Documentation Library

Gamma/Photon

Using Gamma™ with the Photon microGUI®, Version 1.1

Cogent Real-Time Systems, Inc.

August 11, 2010

Gamma/Photon: Using Gamma™ with the Photon microGUI®, Version 1.1

Gamma is the only dynamic language currently available for developing GUIs in the Photon environment. Gamma enhances Photon and PhAB with easy-to-use callbacks, expanded functionality for reusing widgets, and an object-oriented syntax.

Published August 11, 2010
Cogent Real-Time Systems, Inc.
162 Guelph Street, Suite 253
Georgetown, Ontario
Canada, L7G 5X7

Toll Free: 1 (888) 628-2028
Tel: 1 (905) 702-7851
Fax: 1 (905) 702-7850

Information Email: info@cogent.ca
Tech Support Email: support@cogent.ca
Web Site: www.cogent.ca

Revision History

Revision 3.4-1	August 2004 Compatible with Cascade DataHub and Cascade Connect Version 5.0.
Revision 3.3-1	September 2001 Source code compatible across QNX 4 and QNX 6.
Revision 3.2-1	September 2000 Renamed "Gamma/Photon", changed function syntax.
Revision 1.1	March 2000 Edited Widget Classes, expanded Programmer’s Manual.
Revision beta 1.0	February 2000 Expanded existing function references, added class references and Programmer’s Manual.

Copyright, trademark, and software license information.

Copyright Notice

Trademark Notice

END-USER LICENSE AGREEMENT FOR COGENT SOFTWARE

IMPORTANT - READ CAREFULLY: This End-User License Agreement ("EULA") is a legal agreement between you (either an individual or a single entity) and Cogent Real-Time Systems Inc. ("Cogent") of 162 Guelph Street, Suite 253, Georgetown, Ontario, L7G 5X7, Canada (Tel: 905-702-7851, Fax: 905-702-7850), from whom you acquired the Cogent software product(s) ("SOFTWARE PRODUCT" or "SOFTWARE"), either directly from Cogent or through one of Cogent's authorized resellers.
The SOFTWARE PRODUCT includes computer software, any associated media, any printed materials, and any "online" or electronic documentation. By installing, copying or otherwise using the SOFTWARE PRODUCT, you agree to be bound by the terms of this EULA. If you do not agree with the terms of this EULA, Cogent is unwilling to license the SOFTWARE PRODUCT to you. In such event, you may not use or copy the SOFTWARE PRODUCT, and you should promptly contact Cogent for instructions on return of the unused product(s) for a refund.

SOFTWARE PRODUCT LICENSE

The SOFTWARE PRODUCT is protected by copyright laws and copyright treaties, as well as other intellectual property laws and treaties. The SOFTWARE PRODUCT is licensed, not sold.

1. **EVALUATION USE:** This software is distributed as "Free for Evaluation", and with a per-use royalty for Commercial Use, where "Free for Evaluation" means to evaluate Cogent's software and to do exploratory development and "proof of concept" prototyping of software applications, and where "Free for Evaluation" specifically excludes without limitation:
    i. use of the SOFTWARE PRODUCT in a business setting or in support of a business activity,
    ii. development of a system to be used for commercial gain, whether to be sold or to be used within a company, partnership, organization or entity that transacts commercial business,
    iii. the use of the SOFTWARE PRODUCT in a commercial business for any reason other than exploratory development and "proof of concept" prototyping, even if the SOFTWARE PRODUCT is not incorporated into an application or product to be sold,
    iv. the use of the SOFTWARE PRODUCT to enable the use of another application that was developed with the SOFTWARE PRODUCT,
    v. inclusion of the SOFTWARE PRODUCT in a collection of software, whether that collection is sold, given away, or made part of a larger collection.
    vi. inclusion of the SOFTWARE PRODUCT in another product, whether or not that other product is sold, given away, or made part of a larger product.

2. **COMMERCIAL USE:** COMMERCIAL USE is any use that is not specifically defined in this license as EVALUATION USE.

3. **GRANT OF LICENSE:** This EULA covers both COMMERCIAL and EVALUATION USE of the SOFTWARE PRODUCT. Either clause (A) or (B) of this section will apply to you, depending on your actual use of the SOFTWARE PRODUCT. If you have not purchased a license of the SOFTWARE PRODUCT from Cogent or one of Cogent's authorized resellers, then you may not use the product for COMMERCIAL USE.
    A. **GRANT OF LICENSE (EVALUATION USE):** This EULA grants you the following non-exclusive rights when used for EVALUATION purposes:

Software: You may use the SOFTWARE PRODUCT on any number of computers, either stand-alone, or on a network, so long as every use of the SOFTWARE PRODUCT is for EVALUATION USE. You may reproduce the SOFTWARE PRODUCT, but only as reasonably required to install and use it in accordance with this LICENSE or to follow your normal back-up practices.

Subject to the license expressly granted above, you obtain no right, title or interest in or to the SOFTWARE PRODUCT or related documentation, including but not limited to any copyright, patent, trade secret or other proprietary rights therein. All whole or partial copies of the SOFTWARE PRODUCT remain property of Cogent and will be considered part of the SOFTWARE PRODUCT for the purpose of this EULA.

Unless expressly permitted under this EULA or otherwise by Cogent, you will not:

i. use, reproduce, modify, adapt, translate or otherwise transmit the SOFTWARE PRODUCT or related components, in whole or in part;

ii. rent, lease, license, transfer or otherwise provide access to the SOFTWARE PRODUCT or related components;

iii. alter, remove or cover proprietary notices in or on the SOFTWARE PRODUCT, related documentation or storage media;

iv. export the SOFTWARE PRODUCT from the country in which it was provided to you by Cogent or its authorized reseller;

v. use a multi-processor version of the SOFTWARE PRODUCT in a network larger than that for which you have paid the corresponding multi-processor fees;

vi. decompile, disassemble or otherwise attempt or assist others to reverse engineer the SOFTWARE PRODUCT;

vii. circumvent, disable or otherwise render ineffective any demonstration time-outs, locks on functionality or any other restrictions on use in the SOFTWARE PRODUCT;

viii. circumvent, disable or otherwise render ineffective any license verification mechanisms used by the SOFTWARE PRODUCT;

ix. use the SOFTWARE PRODUCT in any application that is intended to create or could, in the event of malfunction or failure, cause personal injury or property damage; or

x. make use of the SOFTWARE PRODUCT for commercial gain, whether directly, indirectly or incidentally.

B. **GRANT OF LICENSE (COMMERCIAL USE):** This EULA grants you the following non-exclusive rights when used for COMMERCIAL purposes:

Software: You may use the SOFTWARE PRODUCT on one computer, or if the SOFTWARE PRODUCT is a multi-processor version - on one node of a network, either: (i) as a development systems for the purpose of creating value-added software applications in accordance with related Cogent documentation; or (ii) as a single run-time copy for use as an integral part of such an application. This includes reproduction and configuration of the SOFTWARE PRODUCT, but only as reasonably required to install and use it in association with your licensed processor or to follow your normal back-up practices.

Storage/Network Use: You may also store or install a copy of the SOFTWARE PRODUCT on one computer to allow your other computers to use the SOFTWARE PRODUCT over an internal network, and distribute the SOFTWARE PRODUCT to your other computers over an internal network. However, you must acquire and dedicate a license for the SOFTWARE PRODUCT for each computer on which the SOFTWARE PRODUCT is used or to which it is distributed. A license for the SOFTWARE PRODUCT may not be shared or used concurrently on different computers.

Subject to the license expressly granted above, you obtain no right, title or interest in or to the SOFTWARE PRODUCT or related documentation, including but not limited to any copyright, patent, trade secret or other proprietary rights therein. All whole or partial copies of the SOFTWARE PRODUCT remain property of Cogent and will be considered part of the SOFTWARE PRODUCT for the purpose of this EULA.

Unless expressly permitted under this EULA or otherwise by Cogent, you will not:

i. use, reproduce, modify, adapt, translate or otherwise transmit the SOFTWARE PRODUCT or related components, in whole or in part;

ii. rent, lease, license, transfer or otherwise provide access to the SOFTWARE PRODUCT or related components;

iii. alter, remove or cover proprietary notices in or on the SOFTWARE PRODUCT, related documentation or storage media;

iv. export the SOFTWARE PRODUCT from the country in which it was provided to you by Cogent or its authorized reseller;

v. use a multi-processor version of the SOFTWARE PRODUCT in a network larger than that for which you have paid the corresponding multi-processor fees;

vi. decompile, disassemble or otherwise attempt or assist others to reverse engineer the SOFTWARE PRODUCT;

vii. circumvent, disable or otherwise render ineffective any demonstration time-outs, locks on functionality or any other restrictions on use in the SOFTWARE PRODUCT;

viii. circumvent, disable or otherwise render ineffective any license verification mechanisms used by the SOFTWARE PRODUCT, or

ix. use the SOFTWARE PRODUCT in any application that is intended to create or could, in the event of malfunction or failure, cause personal injury or property damage.

4. **WARRANTY:** Cogent cannot warrant that the SOFTWARE PRODUCT will function in accordance with related documentation in every combination of hardware platform, software environment and SOFTWARE PRODUCT configuration. You acknowledge that software bugs are likely to be identified when the SOFTWARE PRODUCT is used in your particular application. You therefore accept the responsibility of satisfying yourself that the SOFTWARE PRODUCT is suitable for your intended use. This includes conducting exhaustive testing of your application prior to its initial release and prior to the release of any related hardware or software modifications or enhancements.

   Subject to documentation errors, Cogent warrants to you for a period of ninety (90) days from acceptance of this EULA (as provided above) that the SOFTWARE PRODUCT as delivered by Cogent is capable of performing the functions described in related Cogent user documentation when used on appropriate hardware. Cogent also warrants that any enclosed disk(s) will be free from defects in material and workmanship under normal use for a period of ninety (90) days from acceptance of this EULA. Cogent is not responsible for disk defects that result from accident or abuse. Your sole remedy for any breach of warranty will be either: i) terminate this EULA and receive a refund of any amount paid to Cogent for the SOFTWARE PRODUCT, or ii) to receive a replacement disk.

5. **LIMITATIONS:** Except as expressly warranted above, the SOFTWARE PRODUCT, any related documentation and disks are provided "as is" without other warranties or conditions of any kind, including but not limited to implied warranties of merchantability, fitness for a particular purpose and non-infringement. You assume the entire risk as to the results and performance of the SOFTWARE PRODUCT. Nothing stated in this EULA will imply that the operation of the SOFTWARE PRODUCT will be uninterrupted or error free or that any errors will be corrected. Other written or oral statements by Cogent, its representatives or others do not constitute warranties or conditions of Cogent.

   In no event will Cogent (or its officers, employees, suppliers, distributors, or licensors: collectively "Its Representatives") be liable to you for any indirect, incidental, special or consequential damages whatsoever, including but not limited to loss of revenue, lost or damaged data or other commercial or economic loss, arising out of any breach of this EULA, any use or inability to use the SOFTWARE PRODUCT or any claim made by a third party, even if Cogent (or Its Representatives) have been advised of the possibility of such damage or claim. In no event will the aggregate liability of Cogent (or that of Its Representatives) for any damages or claim, whether in contract, tort or otherwise, exceed the amount paid by you for the SOFTWARE PRODUCT.

   These limitations shall apply whether or not the alleged breach or default is a breach of a fundamental condition or term, or a fundamental breach. Some jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, or certain limitations of implied warranties. Therefore the above limitation may not apply to you.

6. **DESCRIPTION OF OTHER RIGHTS AND LIMITATIONS:**

   Separation of Components. The SOFTWARE PRODUCT is licensed as a single product. Its component parts may not be separated for use on more than one computer.

Termination. Without prejudice to any other rights, Cogent may terminate this EULA if you fail to comply with the terms and conditions of this EULA. In such an event, you must destroy all copies of the SOFTWARE PRODUCT and all of its component parts.

7. **UPGRADES:** If the SOFTWARE PRODUCT is an upgrade from another product, whether from Cogent or another supplier, you may use or transfer the SOFTWARE PRODUCT only in conjunction with that upgrade product, unless you destroy the upgraded product. If the SOFTWARE PRODUCT is an upgrade of a Cogent product, you now may use that upgraded product only in accordance with this EULA. If the SOFTWARE PRODUCT is an upgrade of a component of a package of software programs which you licensed as a single product, the SOFTWARE PRODUCT may be used and transferred only as part of that single product package and may not be separated for use on more than one computer.
8. **COPYRIGHT:** All title and copyrights in and to the SOFTWARE PRODUCT (including but not limited to any images, photographs, animations, video, audio, music, text and ’applets", incorporated into the SOFTWARE PRODUCT), any accompanying printed material, and any copies of the SOFTWARE PRODUCT, are owned by Cogent or its suppliers. You may not copy the printed materials accompanying the SOFTWARE PRODUCT. All rights not specifically granted under this EULA are reserved by Cogent.
9. **PRODUCT SUPPORT:** Cogent has no obligation under this EULA to provide maintenance, support or training.
10. **RESTRICTED RIGHTS:** Use, duplication, or disclosure by the U.S. Government is subject to restrictions as provided in DFARS 227.7202-1(a) and 227.7202-3(a) (1995), DFARS 252.227-7013(c)(1)(ii) (OCT 1988), FAR 12.212(a)(1995), FAR 52.227-19, or FAR 52.227-14 (ALT III), as appropriate. Manufacturer is Cogent Real-Time Systems Inc. 162 Guelph Street, Suite 253, Georgetown, Ontario, L7G 5X7, Canada.
11. **GOVERNING LAW:** This Software License Agreement is governed by the laws of the Province of Ontario, Canada. You irrevocably attorn to the jurisdiction of the courts of the Province of Ontario and agree to commence any litigation that may arise hereunder in the courts located in the Judicial District of Peel, Province of Ontario.

Table of Contents

List of Tables

Chapter 1. Introduction

1.1. Why Gamma and Photon?

Gamma was originally written to develop applications in the QNX operating system. As an alternative to C, Gamma offers fast memory management, an open-event model, and run-time modifiability. Gamma draws upon the power of the QNX operating system in ways that C is not able to.

With the introduction of Photon in 1995, Gamma was expanded with an optional Photon API to take full advantage of and even extend Photon's capabilities. Today Gamma stands out as the only dynamic language available for Photon, offering a wide range of enhancements such as sub-classing of widgets, improved event-handling, highly-flexible callbacks, and code re-usability.

1.2. Assumptions about the Reader

To use Gamma in Photon, you should have some familiarity with Gamma, Photon, and the Photon Application Builder (PhAB). Experience in object-oriented programming will also be useful.

1.3. System Requirements

QNX 6

- QNX 6.1.0 or later.
- Photon 2.

QNX 4

- QNX 4.23A or later.
- Photon 1.14 or later.

1.4. Cogent Product Integration

Cogent products work together to support real-time data connectivity in Windows, Linux, and QNX. They can be dynamically integrated as a group of modules where each module connects to any other module(s) as needed. New modules can be added and existing modules reconfigured or modified, all during run-time. Data in any module of the system can be collected and redistributed to any other module via the Cascade DataHub and Cascade Connect. Communication with field devices is provided by one of several Cogent Device Drivers. Historical records of unlimited size can be maintained and queried with the Cascade Historian, and ASCII text files can be logged with the Cascade TextLogger.

Custom programs written in C or C++ can interface with the system, using the Cogent C API or the DataHub APIs for C++, Java, and .NET. In addition, Cogent's own dynamically-typed object-oriented programming language, Gamma, is fully compatible with all modules. User interfaces can be created in Gamma, which supports Photon in QNX and GTK in Linux.

1.5. Where can I get help?

If you are having problems with a Cogent product, first check the Troubleshooting Guide. If you can't find the answer there, you can contact Cogent Real-Time Systems, Inc. for technical support for any product you have purchased.

- Email: <`support@cogent.ca`>
- Phone: 1-888-628-2028
- Fax: (905) 702-7850

Chapter 2. Classes

2.1. Explanation of Class Reference Pages

Here is a sample class reference page.

A Typical Class Reference Page

`SampleClass` — A one-line description of the class appears here.

Synopsis

Here you will find a class synopsis, including the class name, parent name, and the instance variables.

In all of these reference pages, links to Gamma documentation work for the HTML and QNX Helpviewer editions of this book. Links to QNX Helpviewer documentation work for the QNX Helpviewer edition only.

```
class PtSample ParentClass(Link to Gamma documentation.)
{
    sample_variable_1; //PtSampleVariableOne (Link to QNX Helpviewer.)
    sample_variable_2; //unsigned long (Link to QNX Helpviewer.)
}
```

Base Classes

This shows the base class hierarchy from which this class came. The hierarchy descends from left to right, and is linked to Gamma documentation:

`PtWidget` <-- `PtBasic` <-- `PtContainer` <-- `SampleClass`

Description

Here you will find a more detailed description of the class, although for many classes a complete description is beyond the scope of this document. In those cases, a note and a link will refer you to the Photon documentation.

Instance Variables

There are three kinds of instance variables: (1) those with mutually exclusive constants, (2) those that can take zero or more constants (inclusive), and (3) those with no constants. Examples of each of the three are given below. Those that have constants list the constant names with either a brief description or a one-word title of each constant.

`exclusive_constants_variable` (Linked to QNX Helpviewer when possible.)

A brief description of the variable may appear here.

This instance variable contains exactly one of the following constants:

Constant	Description
`CONSTANT_NAME1`	Constant title or brief description.
`CONSTANT_NAME2`	Constant title or brief description.

`inclusive_constants_variable` (Linked to QNX Helpviewer when possible.)

A brief description of the variable may appear here.

This instance variable may contain zero or more of the following constants:

Constant	Description
`CONSTANT_NAME1`	Constant title or brief description.
`CONSTANT_NAME2`	Constant title or brief description.

`no_constants_variable` (Linked to QNX Helpviewer when possible.)

A brief description of the variable may appear here.

Callbacks

If there are any callbacks associated with this class, their names and a short description will appear here.

Associated Classes

If there are any other classes associated with this class, their names will appear here, linked to the Gamma documentation.

Methods

For the one or two classes that have a method library, their methods will be displayed here, along with their respective syntax, arguments, return value, description and examples.

Convenience Functions

If the class has any convenience functions associated with it, they will appear here. Since multiple functions for one widget generally take the same arguments, all the arguments for the functions are listed first. There there is a brief summary of each function, including return values.

Example

If the class has any example code, it will appear here.

2.2. Specifying Widget Flags

The syntax for handling widget flags in Gamma is somewhat unique. What follows is a brief description of how to set, test, and clear flags on widgets in two different ways. The first is through a non-standard use of the assignment operator, and the second is through method calls. Following these is a short discussion on using bitmasks for flags that are not independent. For more examples, see the tutorial Set, Test, and Clear Widget Flags in the Building Applications - a Tutorial chapter.

2.2.1. Setting Flags

Flags can be set for widget instance variables using the = (assignment) operator and the name of the flag.

The = operator is used in a unique way in Gamma for setting widget flags. It does *NOT* assign a completely new value to the variable. Rather, it adds or removes the value of a *flag* in the variable. The = operator in this context of Gamma has the same effect as the |= operator in C.

For example, to highlight a pane called `Pane1`, you would set the `Pt_HIGHLIGHTED` flag in the `Pane1.flags` variable, as follows.

```
Gamma> Pane1.flags = Pt_HIGHLIGHTED;
256
```

You can set several flags at once as long as you separate them by the | (bitwise or) operator. For example, to set `Pane1` and etch its highlight, you would do this:

```
Gamma> Pane1.flags = Pt_SET | Pt_ETCH_HIGHLIGHT;
514
```

You can also set flags on a widget by assigning the value of the `flags` variable from another widget. When doing this, it is possible to mask one or more of the flags, so that they aren't set on the second widget. This requires conjoining the variable with negated constant(s) using the & (bitwise and) and ~ (bitwise not) operators. For example, to copy the flags from the example above to another pane called `Pane2` without the `Pt_ETCH_HIGHLIGHT` flag or the `Pt_SET` flags on, you could do the following:

```
Gamma> Pane2.flags = Pane1.flags & ~Pt_ETCH_HIGHLIGHT & ~Pt_SET;
256
```

2.2.2. Testing Flags

If you need to check a variable to see if a constant flag is set on it or not, you can test it. Testing is done with the & (bitwise and) operator, the same as in C. However, once the `flags` variable has been conjoined with the constant, that value must be checked to see if it is equal to `0`. This is because in Gamma `0` does not have the logical value of false. You have to make an explicit logical test.

For example, to test `Pane1` and `Pane2` for `Pt_ETCH_HIGHLIGHT`, you could do this:

```
Gamma> (Pane1.flags & Pt_ETCH_HIGHLIGHT) != 0;
t
Gamma> (Pane2.flags & Pt_ETCH_HIGHLIGHT) != 0;
nil
```

Gamma returns `t` if the flag variable is set, or `nil` if it is not set.

2.2.3. Clearing Flags

Flags are cleared using the `cons` function and `nil`. You make a cons cell with the flag as its first element and `nil` as its second element. This has the effect of giving the flag a value of `0`, like using the &= and ~ operators together in C. For example, to clear the `Pt_ETCH_HIGHLIGHT` and `Pt_HIGHLIGHTED` flags from `Pane1`, you would do this:

```
Gamma> Pane1.flags = cons(Pt_ETCH_HIGHLIGHT, nil);
(512)
Gamma> Pane1.flags = cons(Pt_HIGHLIGHTED, nil);
(256)
```

```
Gamma> (Pane1.flags & (Pt_ETCH_HIGHLIGHT | Pt_HIGHLIGHT)) != 0;
nil
```

Incidentally, the `cons` function can also be used to set flags. Just cons the flag to `t` instead of `nil`. For example, to set the `Pt_SET` flag on `Pane2`, you could do this:

```
Gamma> Pane2.flags = cons(Pt_SET,t);
(2 . t)
Gamma> (Pane2.flags & Pt_SET) != 0;
t
```

2.2.4. Using Widget Method Calls

As an alternative to using the assignment operator, you can set, test, and clear instance variable flags on widgets using the `SetBit`, `TestBit`, and `ClearBit` methods of `PtWidget`, which is the parent widget for all of the Widget Classes. (See PtWidget for more information.) These methods are in the `PhotonWidgets.lsp` library, so you must make a call to **`require_lisp("PhotonWidgets.lsp")`** before using them.

Continuing with the example of `Pane1` from above, you could set, test, and clear its `Pt_HIGHLIGHTED` flag as follows:

```
Gamma> require_lisp("PhotonWidgets.lsp");
"<pathname>PhotonWidgets.lsp"
Gamma> Pane1.SetBit(#flags, Pt_HIGHLIGHTED);
256
Gamma> Pane1.TestBit(#flags, Pt_HIGHLIGHTED);
t
Gamma> Pane1.ClearBit(#flags, Pt_HIGHLIGHTED);
(256)
Gamma> Pane1.TestBit(#flags, Pt_HIGHLIGHTED);
nil
```

Using these methods and/or the assignment operator as explained here, you should have no problem setting flags on most widgets.

2.2.5. Using Bitmasks and cons

Some flags are exclusive, meaning there can be only one of a group of them set at any given time. Setting one of these flags requires that any others in the group be unset first, but without unsetting those flags on the variable that are not part of the group. The easy way to do this is to use bitmasks and the `cons` function.

A bitmask consists of all the exclusive flags in the group ORed together. To set a flag, you simply `cons` it to the bitmask. For example, the `RtTrend` widget has a variable named `rttrend_flags`, with three flags that must be set exclusively: `Rt_GRID_IS_TRANSLUCENT`, `Rt_GRID_ABOVE_TRENDS`, and `Rt_TRENDS_ABOVE_GRID`. To set, for example, `Rt_GRID_ABOVE_TRENDS`, you would do this:

```
Gamma>  mask = Rt_GRID_IS_TRANSLUCENT
           | Rt_GRID_ABOVE_TRENDS
           | Rt_TRENDS_ABOVE_GRID;
1792
Gamma> trend.rttrend_flags = cons(Rt_GRID_ABOVE_TRENDS, mask);
(256 . 1792)
Gamma> (trend.rttrend_flags & Rt_GRID_ABOVE_TRENDS) != 0;
t
```

Doing this:

```
cons(flags, mask)
```

is exactly equivalent to doing this:

```
(original & ~mask) | flags
```

where *original* is the original value of the variable. To illustrate what happens at bitwise level, let's look at two flags on an imaginary variable:

```
Flag 1       0010000
Flag 2       0001000
Bitmask      0011000
```

Suppose Flags 1 and 2 must be set exclusively from each other. In the example below, the variable has a few flags set already, including Flag 1. We want set Flag 1 OFF and Flag 2 ON. Using the Bitmask, we AND the original flags of the variable to the inverse of the bitmask, turning both flags OFF. Then we OR the result to Flag 2 to turn Flag 2 ON.

```
original:       10100010   Flag 1 and other flags are ON, Flag 2 is OFF.
AND ~Bitmask    11001111
                --------
results:        10000010   Flag 1 is now OFF.


                10000010   The other flags are still ON.
OR Flag 2       00010000
                --------
final result:   10010010   Flag 2 is now ON as well.
```

2.2.6. Using cons: a Summary

The cons can be used in Gamma in different ways for different purposes. The following table gives a summary of how cons is used for handling widget flags.

Table 2-1. Using cons with widget flags.

Gamma Syntax	Meaning
widget.variable = flags;	widget.variable = (original | flags)
widget.variable = cons(flags, t);	widget.variable = (original | flags)
widget.variable = cons(flags, nil);	widget.variable = (original & ~flags)
widget.variable = cons(flags, mask);	widget.variable = (original & ~mask) | flags

2.3. Specifying Colors

Colors in Gamma are represented by 24 bit hexadecimal numbers ranging from 0x000000 to 0xffffff. The first 8 bits correspond to the red value, the next 8 to the green value, and the last 8 to the blue value. For example, 0x00ff00 would be pure green.

Most people find it easier to think in decimal notation, though, and the red, green, and blue colors are often assigned values from 0 - 255, representing a scale of intensity from black to full color. Gamma allows for such red/green/blue notation for colors with the PgRGB function. It converts colors expressed in red/blue/green decimal notation to a single hexadecimal number. For example:

```
Gamma> lightblue = PgRGB(185,223,240);
0xb9dff0
```

To assign this color to a PtWindow named win, and then test the variable, we would do this:

```
Gamma> win.color = PgRGB(185,223,240);
0xb9dff0;
Gamma> win.color;
```

```
0xb9dff0
```

In this example, the red value of `185` is `0xb9`, the green value of `223` is `0xdf`, and the blue value of `240` is `0xf0`. To demonstrate this, we can call the functions `PgRedValue`, `PgGreenValue`, and `PgBlueValue` on the win.color variable to return the values for their respective bits:

```
Gamma> PgRedValue(win.color);
0xb9
Gamma> PgGreenValue(win.color);
0xdf
Gamma> PgBluePValue(win.color);
0xf0
```

Transparent Although we said that colors in Gamma are represented with 24 bits, internally they actually are stored as 32 bit numbers. The first 8 bits are always ignored, however, except in one case--the non-color: transparent, which is represented as `0xffffffff` (all 32 bits are on).

Chapter 3. A Few Precautions

Due to Gamma's dynamic structure, and the extra demands of interfacing with the Photon environment, there are a few precautions that a prudent programmer should take when coding. Failure to observe these precautions could lead to unpredictable behavior and even crash the Gamma engine.

Instance arguments. Gamma functions that take instances as arguments check their type to confirm that they are instances. However, in the interests of speed and efficiency, they do not check that the instance belongs to the class that will work with that particular function. You as the programmer are responsible for making sure an instance of the correct class has been passed. Otherwise you may see unexpected behavior in your program.

Callbacks. Throwing an error in a callback can cause failures in subsequent Photon calls. You can protect your code from callback errors using `PtProtectCallbacks`, but you will receive only the first of any error messages.

`PtCallbackInfo`. This is a special class that overlays several other classes and holds information about their corresponding callbacks. It should only be accessed with a valid instance variable, from one class at a time. Attempting to extract all information at once from this class with a call to, say, `princ` could crash the Gamma engine.

`PtInit`. This function must be called before using any Photon functions, as it initializes the widget library.

`init_ipc`. This function must be called before any interprocess communication is attempted, as it initializes the internal IPC data structures used by Gamma.

Chapter 4. Building Applications - a Tutorial

Here is a graduated tutorial that introduces some of the key features of Gamma, such as creating widgets, attaching callbacks, setting flags, and manipulating widget files for reuse. The programs and any associated widget files are in the distribution. The tutorial programs are numbered t_01 through t_11. You can run them from the command line by using the command **phgamma** followed by the name of the tutorial. For example, type **`phgamma t_04`** to run Tutorial 4: Load a Window Created in PhAB.

4.1. Create a Window

```
#!/usr/cogent/bin/phgamma

/*
 * This example demonstrates how to create a Photon application in Gamma.
 * Since the Photon widget set is mapped as OOP classes in Gamma, any
 * Photon widget can be created using the 'new' function and then
 * manipulated by simply modifying its instance variables.
 *
 * This file is best viewed with a tab width of 4.
 *
 * First, we load some Photon Widget convenience functions to make our
 * job easier.  In this case, we are loading support for the SetDim,
 * SetPos and other functions which we use in this and later examples.
 */
require_lisp("PhotonWidgets");

/*
 * Initiate the graphics session with the Photon window manager.  This
 * must always be done before any Photon call can be made.
 */
PtInit(nil);

/*
 * Create a new instance of a PtWindow object.  You can look up PtWindow
 * in the Reference Manual, which is also linked to the online help for
 * Photon.  Gamma uses the same resources and constants as Photon widgets,
 * except that resource names are in lower case, and have the Pt_ARG_
 * prefix removed.  All constants are named identically to the ones named
 * in the Photon documentation.
 */
window = new(PtWindow);

/*
 * Set the width and height of the window.
 */
window.SetDim(300,250);

/*
 * Set the fill color of the window.
 */
window.fill_color = PgRGB(240,220,220);

/*
 * Realize the window widget on the screen.  A widget is not visible
 * until it has been explicitly realized, or until its container is
 * realized.
 */
PtRealizeWidget(window);

/*
 * Start an infinite event loop to handle Photon events. This keeps the
 * window open until the program exits (such as when Alt_F4 is pressed).
 */
PtMainLoop();
```

4.2. Add an Exit Button

```
#!/usr/cogent/bin/phgamma

/*
 * This example adds an exit button to the window created in the last
 * example, illustrating how easy it is to attach a callback.
 */

require_lisp("PhotonWidgets");
PtInit(nil);

window = new(PtWindow);
window.SetDim (300,250);
window.fill_color = PgRGB (240,220,220);

/*
 * Create a PtButton Widget and set resources.
 */
but = new(PtButton);
but.SetPos(130,205);
but.text_string = " Exit ";

/*
 * Attach a callback to the button that will exit the program when
 * the button is pressed and released.
 */
PtAttachCallback(but,Pt_CB_ACTIVATE,#exit_program(-1));

PtRealizeWidget(window);
PtMainLoop();
```

4.3. Add Functionality with Callbacks

```
#!/usr/cogent/bin/phgamma

/*
 * This example adds a pane and a text-entry box to the last example.
 * The pane displays the color that corrresponds to the hexidecimal
 * numerical value entered in the text-entry box.
 *
 * This file is best viewed with a tab width of 4.
 */

require_lisp("PhotonWidgets");
PtInit(nil);

window = new(PtWindow);
window.SetDim (300,250);
window.fill_color = PgRGB (240,220,220);

/*
 * Make a new pane, and set its area using the SetArea function, which
 * combines SetPos and SetDim.
 */
pane = new(PtPane);
pane.SetArea (50,50,200,100);

/*
 * Set the window to be the parent for the next widgets.
 */
```

```
PtSetParentWidget(window);

/*
 * Make a new text box, set its area, and attach callbacks. The first
 * callback parses the entry string, and the second one changes the
 * color of the pane.
 */
entrybox = new(PtText);
entrybox.SetArea (115,160,60,25);
PtAttachCallback(entrybox,Pt_CB_TEXT_CHANGED,
                 #entry = parse_string(entrybox.text_string));
PtAttachCallback(entrybox,Pt_CB_ACTIVATE,#pane.fill_color = entry);

but = new(PtButton);
but.SetPos(130,205);
but.text_string = "Exit";
PtAttachCallback(but,Pt_CB_ACTIVATE,#exit_program(-1));

PtRealizeWidget(window);
PtMainLoop();
```

4.4. Load a Window Created in PhAB

```
#!/usr/cogent/bin/phgamma

/* This example is the first of three ways to incorporate a
 * PhAB-created GUI into a Gamma program.  It loads from PhAB a window
 * similar to the one created in the last example, and attaches
 * callbacks to the widgets.  This illustrates an advantage of using
 * Gamma and PhAB together.  Complex widgets can be created and
 * changed quickly in PhAB, while callbacks and other code can be
 * attached easily in Gamma.*/

require_lisp("PhotonWidgets.lsp");
require_lisp("PhabTemplate.lsp");
PtInit(nil);

/*
 * Load the window that was previously created in PhAB.
 */
PhabLoad (string(_os_, "-WidgetFiles/wgt/colortest.wgtw"));

/*
 * Attach callbacks.
 */
PtAttachCallback(entrybox,Pt_CB_TEXT_CHANGED,
                 #entry = parse_string(entrybox.text_string));
PtAttachCallback(entrybox,Pt_CB_ACTIVATE,#colorpane.fill_color = entry);
PtAttachCallback(exitbutcolortest,Pt_CB_ACTIVATE,#exit_program(-1));

PtRealizeWidget(colortest);
PtMainLoop();
```

4.5. Read a Widget File and Create its Widgets

```
#!/usr/cogent/bin/phgamma

/* This example is the second of three ways to incorporate a
 * PhAB-created GUI into a Gamma program.  It produces the same output
 * as the previous example, but instead of using the PhabLoad()
 * function, it calls PhabReadWidgetFile() and PhabCreateWidgets().
 * This allows for naming the widgets locally, and independently of
 * their original names, unlike PhabLoad(), which assigns global
 * variables that correspond to widget names.  */

require_lisp("PhotonWidgets.lsp");
require_lisp("PhabTemplate.lsp");
PtInit(nil);

/*
 * Read the widget file.
 */
wfile = PhabReadWidgetFile (string(_os_, "-WidgetFiles/wgt/colortest.wgtw"));

/*
 * Create widgets from the widget file that was read.
 */
window = PhabCreateWidgets(wfile, nil, nil);

/*
 * Find the widgets to be used, and assign them new names.
 */
ct = PhabLookupWidget(window,#colortest,nil);
eb = PhabLookupWidget(window,#entrybox,nil);
pn = PhabLookupWidget(window,#colorpane,nil);
bt = PhabLookupWidget(window,#exitbutcolortest,nil);

PtAttachCallback(eb,Pt_CB_TEXT_CHANGED,
                 #entry = parse_string(eb.text_string));
PtAttachCallback(eb,Pt_CB_ACTIVATE,#pn.fill_color = entry);
PtAttachCallback(bt,Pt_CB_ACTIVATE,#exit_program(-1));

PtRealizeWidget(ct);
PtMainLoop();
```

4.6. Read a Widget File and Assign its Widgets to a Class

```
#!/usr/cogent/bin/phgamma

/* This example is the third of three ways to incorporate a
 * PhAB-created GUI in a Gamma program.  It uses PhabAttachWidgets()
 * to assign the widgets to one global class, with each widget as an
 * instance of the class.  For most programming situations, this is
 * probably the best of the three alternatives.  */

require_lisp("PhotonWidgets.lsp");
require_lisp("PhabTemplate.lsp");
PtInit(nil);

/* Create the class.*/
class ColorTest
{
  window;
}

/* Attach the widgets to the class. */
PhabAttachWidgets (ColorTest, string (_os_, "-WidgetFiles/wgt/colortest.wgtw"));
```

```
/* Set up the constructor to instantiate the widgets inside the window. */
method ColorTest.constructor ()
{
  .window = PhabRoot(.PhabInstantiate (t));
}

/* Create a new instance of the class and realize it. */
cwin = new (ColorTest);
PtRealizeWidget (cwin.window);

/* Assign callbacks by calling the widgets as instances of
 * the ColorTest class.*/
PtAttachCallback(cwin.entrybox,Pt_CB_TEXT_CHANGED,
                 #entry = parse_string(cwin.entrybox.text_string));
PtAttachCallback(cwin.entrybox,Pt_CB_ACTIVATE,#cwin.colorpane.fill_color = entry);
PtAttachCallback(cwin.exitbutcolortest,Pt_CB_ACTIVATE,#exit_program(-1));

PtMainLoop();
```

4.7. Set, Test, and Clear Widget Flags

```
#!/usr/cogent/bin/phgamma

/*
 * This example demonstrates setting, testing, and clearing widget
 * flags. You can compare the results by viewing the original file:
 * QNX4-WidgetFiles/wgt/setflags.wgtw in PhAB, QNX 4, or
 * QNX6-WidgetFiles/wgt/setflags.wgtw in the Builder, QNX 6.
 *
 * Start by loading the necessary Photon and PhAB files, and initializing
 * Photon.
 */
require_lisp("PhotonWidgets.lsp");
require_lisp("PhabTemplate.lsp");
PtInit(nil);

/*
 * Read the widget file, create it, and assign symbols to the widgets
 * to be used.
 */
wfile = PhabReadWidgetFile (string(_os_, "-WidgetFiles/wgt/setflags.wgtw"));
window = PhabCreateWidgets(wfile, nil, nil);

setflags = PhabLookupWidget(window,#setflags,nil);
p1 = PhabLookupWidget(window,#Pane1,nil);
p2 = PhabLookupWidget(window,#Pane2,nil);
p3 = PhabLookupWidget(window,#Pane3,nil);
p4 = PhabLookupWidget(window,#Pane4,nil);
p5 = PhabLookupWidget(window,#Pane5,nil);
p6 = PhabLookupWidget(window,#Pane6,nil);
p7 = PhabLookupWidget(window,#Pane7,nil);
p8 = PhabLookupWidget(window,#Pane8,nil);
p9 = PhabLookupWidget(window,#Pane9,nil);

/*
 * Set a single flag; set multiple flags using the | bitwise operator;
 * and copy and set multiple flags using the & and ~ operators for masking.
 */
p1.flags = Pt_HIGHLIGHTED;
p2.flags = Pt_HIGHLIGHTED | Pt_SET | Pt_ETCH_HIGHLIGHT;
p3.flags = p2.flags & ~Pt_ETCH_HIGHLIGHT;
```

```
/*
 * Test widgets, whose flags were originally set in PhAB, for flags
 * set/cleared status using the & bitwise operator. Print the results.
 */
princ("\nPane 4 flags are: ", p4.flags,"\n");
test1 = hex(p4.flags & Pt_HIGHLIGHTED);
test2 = hex(p4.flags & Pt_SET);
test3 = hex(p4.flags & Pt_ETCH_HIGHLIGHT);
princ("Test 1 results: ", test1, "\n");
princ("Test 2 results: ", test2, "\n");
princ("Test 3 results: ", test3, "\n");

princ("\nPane 5 flags are: ", p5.flags,"\n");
test1 = hex(p5.flags & Pt_HIGHLIGHTED);
test2 = hex(p5.flags & Pt_SET);
test3 = hex(p5.flags & Pt_ETCH_HIGHLIGHT);
princ("Test 1 results: ", test1, "\n");
princ("Test 2 results: ", test2, "\n");
princ("Test 3 results: ", test3, "\n");

princ("\nPane 6 flags are: ", p6.flags,"\n");
test1 = hex(p6.flags & Pt_HIGHLIGHTED);
test2 = hex(p6.flags & Pt_SET);
test3 = hex(p6.flags & Pt_ETCH_HIGHLIGHT);
princ("Test 1 results: ", test1, "\n");
princ("Test 2 results: ", test2, "\n");
princ("Test 3 results: ", test3, "\n\n");

/*
 * Clear certain flags, using the cons() function, from widgets
 * whose flags were originally set in PhAB.  Before the flags were
 * cleared, p7.flags were the same as p4.flags; p8.flags were
 * equal to p5.flags; and p9.flags were equal to p6.flags.
 */
p7.flags = cons(Pt_HIGHLIGHTED,nil);
p8.flags = cons(Pt_SET,nil);
p9.flags = cons(Pt_SET,nil);
p9.flags = cons(Pt_HIGHLIGHTED,nil);
p9.flags = cons(Pt_ETCH_HIGHLIGHT,nil);

/*
 * Test, set, and clear flags using PtWidget method calls.
 */
princ("Pane 1 flags: ",p1.flags,"\n");
princ("Highlighted flag set? ",
      p1.TestBit(#flags,Pt_HIGHLIGHTED),"\n\n");

p1.SetBit(#flags,Pt_ETCH_HIGHLIGHT | Pt_SET);
princ("Pane 1 flags: ",p1.flags,"\n");
princ("Etch highlight flag set? ",
      p1.TestBit(#flags,Pt_ETCH_HIGHLIGHT),"\n");
princ("Set flag set? ",p1.TestBit(#flags,Pt_SET),"\n\n");

p1.ClearBit(#flags,Pt_ETCH_HIGHLIGHT | Pt_SET);
princ("Pane 1 flags: ",p1.flags,"\n");
princ("Etch highlight flag still set? ",
      p1.TestBit(#flags,Pt_ETCH_HIGHLIGHT),"\n");
princ("Set flag still set? ",p1.TestBit(#flags,Pt_SET),"\n\n");

PtRealizeWidget(setflags);
PtMainLoop();
```

4.8. Read and Print a Widget File

```
#!/usr/cogent/bin/phgamma

/*
 * This example reads and prints a widget file locating each widget and
 * printing its level, type, and name on one line, followed by its resources.
 * The program reads a file name from the command line, using the argument
 * (cadr(argv)). Our test file is a window that contains one button and
 * two panes, each of which also contain buttons. Enter its name on the
 * command line as: QNX4-WidgetFiles/wgt/readtestfile.wgtw for QNX 4, or
 * QNX6-WidgetFiles/wgt/readtestfile.wgtw for QNX 6.
 */

PtInit(nil);

/*
 * Define some PhabGet- functions. (These are scheduled to be added as
 * library functions in the near future).
 */
PhabGetTop = car;
PhabGetRoot = car;
PhabGetChildren = cdr;
PhabGetLevel = car;
PhabGetType = cadr;
PhabGetName = caddr;
function PhabGetResources (wgt)
     {
       caddr(cdr(wgt));
     }

require_lisp("PhotonWidgets");
require_lisp("PhabTemplate");

/* Check to see if there is a command-line argument. */
if (cadr(argv))
{
  /*
   * Read and print the complete widget file definition.
   */
  defs = PhabReadWidgetFile(cadr(argv));
  pretty_princ("The widget definition is:\n", defs, "\n\n");

  tree = PhabGetTop(defs);

  /*
   * Make a function that deconstructs the widget file definition,
   * extracting each widget definition in turn, and identifying
   * each part of each definition.
   */
  function print_tree(tree)
 {
   local root = PhabGetRoot(tree);
   local level = PhabGetLevel(root);
   local type = PhabGetType(root);
   local name = PhabGetName(root);
   local resources = PhabGetResources(root);

   princ("The ", name, " widget is type ", type,
   " at level ", level, ". \nIts resources are: \n");
   pretty_princ(resources, "\n");

   local children = PhabGetChildren(tree);
   if (children)
   {
  pretty_princ("Its children are:\n", children, "\n\n");
   }
   else princ("It has no children.\n\n");
```

```
   with child in children do {
  print_tree(child);
   }
 }

  /*
   * Call the function, then create and realize the widgets as an illustration.
   */
  print_tree(tree);

  NewWindow = car(car(PhabCreateWidgets(defs, nil, nil)));
  NewWindow.SetPos(250,50);

  PtRealizeWidget(NewWindow);

  PtMainLoop();
}

else princ("Please enter a command-line argument, such as: \n",
     "QNX4-WidgetFiles/wgt/readtestfile.wgtw in QNX 4, or\n",
     "QNX6-WidgetFiles/wgt/readtestfile.wgtw in QNX 6.\n");
```

4.9. Extract Certain Widgets

```
#!/usr/cogent/bin/phgamma

/*
This example reads a widget file that contains 8 buttons with .gif
images on them.  It creates the widgets and puts them into two
windows.  The first window contains all of the widgets, while the
second window contains a widget of your choice, which you can enter as
the third argument on the command line. To run this example, you must
be in a directory that can access the .wgt file. Type the 2 arguments
buttons and [button name] at the shell prompt, where [button name]
can be one of: Go, Stop, Pause, Done, Increase, Decrease, Left or Right.
*/

PtInit(nil);

/*
 * Create some convenience functions.
 */
PhabGetTop = car;
PhabGetRoot = car;
PhabGetChildren = cdr;
PhabGetLevel = car;
PhabGetType = cadr;
PhabGetName = caddr;

require_lisp("PhotonWidgets.lsp");
require_lisp("PhabTemplate.lsp");

/*
 * Read the widget definitions, and access them.
 */
file = string(_os_, "-WidgetFiles/wgt/buttons.wgtw");
defs = PhabReadWidgetFile(file);
tree = PhabGetTop(defs);
children = PhabGetChildren(tree);

/*
 * Make a function to index the buttons by number.
 */
```

```
function index_buttons(buttons)
{
 local index = 1;
 indexlist = list();

 with button in buttons do
 {
  local name = PhabGetName(car(button));
  indexlist = cons(list(name, index), indexlist);
  index ++;
 }
 indexlist;
}

/*
 * Make a function to look up a requested button.
 */
function lookup (name, list)
{
 car(assoc(name, list));
}

/*
 * Make a new window, create all the buttons, and display them
 * in the window.
 */
win = new(PtWindow);
win.SetArea(100,50,120,500);
for (i=0;i<8;i++)
{
 eachbut = car(car(PhabCreateWidgets(list(car(nth_cdr(children,i)))
  ,nil,nil)));
 if (eachbut.name == #Done)
   PtAttachCallback(eachbut, Pt_CB_ACTIVATE,#exit_program(-1));
 eachbut.SetPos(25, (i * 60));
 PtRealizeWidget(eachbut);
 PtExtentWidget(win);
}

/*
 * Find the requested button.
 */
buttonlist = index_buttons(children);
if (cadr(argv))
  argument = cadr(argv);
else argument = "Go";
found = lookup(symbol(argument), buttonlist);
index = cadr(found) - 1;

/*
 * Make a new window, create the requested button, and display it.
 */
win2 = new(PtWindow);
win2.SetArea(250,50,120,120);
anybut = car(car(PhabCreateWidgets(list(car(nth_cdr(children,index)))
 ,nil,nil)));
anybut.SetPos(20,40);

princ("Choose another button by using one of:\n",
   "Go, Stop, Pause, Done, Increase, Decrease, Left or Right\n",
   "as a command-line argument.\n");

PtRealizeWidget(win);
PtMainLoop();
```

4.10. Select and Recreate Any Widget

```
#!/usr/cogent/bin/phgamma

/*
 * This example is similar to the previous one, in that it pulls
 * specified widgets out of an existing widget, named readtestfile.wgtw.
 * Enter the name for the widget of your choice on the command
 * line.  The possible names are: MyTestFile (brings the whole
 * widget), MainWindowButton, PaneOne, PaneOneButtonA,
 * PaneOneButtonB, PaneTwo, and PaneTwoButtonC.
 */

PtInit(nil);
require_lisp("PhotonWidgets");
require_lisp("PhabTemplate");

/*
 * Read the widget definitions, and access them.
 */
file = string(_os_, "-WidgetFiles/wgt/readtestfile.wgtw");
defs = PhabReadWidgetFile(file);
treewin = car(defs);

/*
 * Make a function that walks a widget tree and finds the widget
 * you want. (The same as in the previous example.)
 */
function PhabLookupTree(tree,name)
  {
    local return = nil;
    local trees;
    if (caddr(car(tree)) == name)
      return = tree;
    else
      {
 for (trees = cdr(tree); trees && !return; trees = cdr(trees)){
    return = PhabLookupTree(car(trees),name);
  }
      }
    return;
  }

/*
 * Make a new window to hold the widget.
 */
win = new(PtWindow);
win.SetDim(500,450);

exitbut = new(PtButton);
exitbut.SetPos(220, 420);
exitbut.text_string = " Exit ";
PtAttachCallback(exitbut,Pt_CB_ACTIVATE,#exit_program(-1));


if (cadr(argv))
{
  /*
   * Call the PhabLookupTree function on a widget from the command line.
   */
  read_name = symbol(cadr(argv));
  wgtdef = PhabLookupTree(treewin,read_name);
  /*
   * Transform the widget tree into a widget definition.
   */
  wgtdef = cons(wgtdef,nil);

  /*
   * Create the widget.
```

```
   */
  wgt = PhabCreateWidgets(wgtdef,nil,nil);

  /*
   * Realize the widget by realizing the window.
   */
  PtRealizeWidget(win);
  PtMainLoop();
}

else princ("\nYou must enter one of the following as a command-line argument: \n",
     "MyTestFile (brings the whole widget), MainWindowButton, PaneOne, \n",
     "PaneOneButtonA, PaneOneButtonB, PaneTwo, and PaneTwoButtonC\n\n");
```

4.11. Print a Widget Tree Summary

```
#!/usr/cogent/bin/phgamma

/*
 * This example reads a widget file and prints the widget level,
 * type, and name in hierarchical order. You can use the test file
 * QNX4-WidgetFiles/wgt/colortest.wgtw for QNX 4 or
 * QNX6-WidgetFiles/wgt/colortest.wgtw for QNX 6 by entering its
 * name as the first argument on the command line.
 */

PtInit(nil);
require_lisp("PhotonWidgets.lsp");
require_lisp("PhabTemplate.lsp");

if (cadr(argv))
{
  /*
   * Read the widget definitions, and access them.
   */
  file = string(cadr(argv));
  defs = PhabReadWidgetFile(file);
  treewin = car(defs);

  /*
   * Make a function that walks a widget tree, and finds
   * and prints the level, name, and type of its widgets.
   */
  function PhabInfoTree(tree)
 {
   local return = nil;
   local trees;
   if (car(tree))
  princ(caar(tree)," ",cadar(tree),"\t\t ",caddr(car(tree)),"\n");
   for (trees = cdr(tree); trees; trees = cdr(trees)){
  return = PhabInfoTree(car(trees));
   }
   return;
 }

  /*
   * Call the function.
   */
  PhabInfoTree(treewin);
}
```

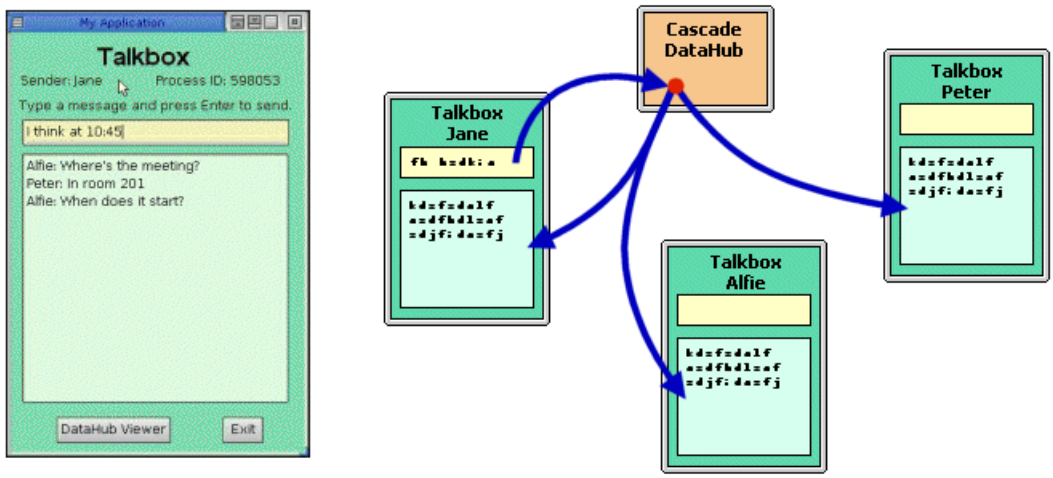
```
else princ("Please enter a command-line argument, such as: \n",
     "QNX4-WidgetFiles/wgt/colortest.wgtw in QNX 4, or\n",
     "QNX6-WidgetFiles/wgt/colortest.wgtw in QNX 6.\n");
```

Chapter 5. Integrating with Cogent Software

5.1. Callbacks and Cascade DataHub Points

Gamma is an integral part of the rest of Cogent's software. Using Gamma with Photon gives you a way to create GUIs, and then join them quickly and seamlessly with other Cogent tools, such as the Cascade DataHub. Using all these tools together, you can create powerful and reliable applications with substantially less work than it would take in C.

Here is an example, written with about 50 lines of code, that creates a "Talk Box", a simple communications device that uses the Cascade DataHub. It works by writing to and reading a single data point on the Cascade DataHub. As the picture illustrates, the text entered on any Talk Box is sent to every other Talk Box that has access to the same datahub point.



Since this program uses the Cascade DataHub, you must ensure that it is running. If not, start it with this command issued at a shell prompt:

```
[sh]$  datahub
```

When the Cascade DataHub is running, enter the command for Talk Box followed by a point name argument, like this:

$ **phgamma talkbox.g *Name* &**

Gamma will then run the following code, starting up a Talk Box. To start more Talk Boxes, enter the same command again, with a different *Name*. Each Talk Box shows its process ID number in the top right-hand corner. To view the Cascade DataHub, press the DataHub Viewer button on any Talk Box.

```
#!/usr/cogent/bin/phgamma

/*
 * This example loads a Talk Box with a text-entry line for writing,
 * and a multi-text field for reading.  When you enter text, it is
 * sent to the datahub, and then appended to the bottom of the text
 * in the multi-text field for reading.
 *
 * This program should be called from the command line like this:
 *
 * > gamma talkbox.g <Name> &
 *
 * Where <Name> can be any name.  If no name is provided the program
 * will assign the process ID number as the name.
 *
```

```
 * First, load the necessary Photon and PhAB files, and initialize Photon.
 */

require_lisp("PhotonWidgets.lsp");
require_lisp("PhabTemplate.lsp");
PtInit(nil);

/*
 * Then create a unique name for this Talk Box, based on its node
 * and process ID numbers. This allows multiple Talk Boxes to run
 * simultaneously.
 */
uniq_name = string(getnid(), ".", getpid());

/*
 * Initialize IPC, read the point name from the command line, and
 * register to receive all changes made to the point.
 */
init_ipc(uniq_name, uniq_name);
dhpoint = "";
register_point(#dhpoint);
if (cadr(argv))
  name = cadr(argv);
else name = string(getpid());
princ("Name: ", name, "\n");

/*
 * Read the widget file, create it, and assign symbols to the widgets
 * to be used.
 */
wfile = PhabReadWidgetFile (string(_os_, "-WidgetFiles/wgt/talkbox.wgtw"));
window = PhabCreateWidgets(wfile, nil, nil);

tlkbx = PhabLookupWidget(window,#talkbox,nil);
enttxt = PhabLookupWidget(window,#EntryText,nil);
distxt= PhabLookupWidget(window,#DisplayText,nil);
dhviewbut = PhabLookupWidget(window,#DHviewButton,nil);
sendlab = PhabLookupWidget(window,#SendLabel,nil);
idlab = PhabLookupWidget(window,#IDLabel,nil);
ebut = PhabLookupWidget(window,#TalkboxExitButton,nil);
PtAttachCallback(ebut, Pt_CB_ACTIVATE, #exit_program(1));

/*
 * Make the reader window display-only.
 */
distxt.text_flags = cons(Pt_EDITABLE, nil);

/*
 * Create a function to write the entered text to the datahub
 * and then delete it from the entry widget.
 */
function entry_cb(entry, dhpoint, name, text)
{
  write_point(#dhpoint, string(name, ": ", enttxt.text_string));
  entry.text_string = "";
}

/* Attach a callback to the above function, and another callback
 * to bring up the datahub viewer when that button is pushed.
 */
PtAttachCallback(enttxt, Pt_CB_ACTIVATE, ‘entry_cb(@enttxt, @dhpoint, @name, @distxt));
PtAttachCallback(dhviewbut, Pt_CB_ACTIVATE, #system("phdhview &"));

/*
 * Create a function to add new text to the display and scroll the
 * display to show the text properly.
 */
function display_txt(new_text, wgt)
```

```
{
  wgt.text_string = string(wgt.text_string, new_text, "\n");
  wgt.multitext_y_scroll_pos = wgt.multitext_num_lines + 2;
}

/* Add an exception function (the display_txt() function) to the
 * datahub point so that when the value changes as a result of other
 * Talkboxes, the text will get displayed.
 */
add_exception_function(#dhpoint, ‘display_txt(dhpoint, @distxt));

/* Add an echo function (the display_txt() function) to the
 * datahub point so that when the value changes as a result of this
 * Talkbox, the text will get displayed.  The Gamma substr() function
 * is used to remove the name from the text message, and the Gamma
 * string() function is used to add the string "Me: " to the beginning
 * of the substring.
 */
add_echo_function(#dhpoint,
    ‘display_txt(string("Me: ", substr(dhpoint, (strlen(name) + 2), -1)),
          @distxt));

/*
 * Have the labels show the sender and PID number.
 */
sendlab.text_string = string("Sender: ", name);
idlab.text_string = string("Process ID: ", getpid());

PtRealizeWidget(tlkbx);
//PtMainLoop();
while(t) next_event();
```

Chapter 6. Sample Code and Cool Stuff

6.1. Customizable Keypad

Here is a customizable keypad class that lets you make keypads with any number of push-button keys in any arrangement, with whatever symbols you choose to assign to them. You can include a display line for entered text, and include special keys such as Shift, Clear, Delete, Enter, and so on, or define your own function keys.

The sample code loads and displays two example keypads, one a numeric keypad with English-language key labels, and the other alpha-numeric with French-language key labels. The definition files used to create the two keypads are shown in the Sample Keypad Definition Files section of this chapter. The source code of the Keypad class itself is given in The Keypad Class section.

6.1.1. Create and Display Two Keypads

```
#!/usr/cogent/bin/phgamma

require_lisp("PhotonWidgets");
require_lisp("const/Filesys.lsp");
require ("keypad.g");
require ("utility.g");

PtInit(nil);

window = new(PtWindow);
window.SetDim (700,380);
window.fill_color = PgRGB (250,200,180);

class USnumberKeypad Keypad
{
}

method USnumberKeypad.init(parent)
{
  .LoadDefinition("number_en_US.kdef", parent);
  .BlockAllWindows(t);
}

class FRalphaKeypad Keypad
{
}

method FRalphaKeypad.init(parent)
{
  .LoadDefinition("alpha_fr_FR.kdef", parent);
  .BlockAllWindows(t);
}

PtSetParentWidget(window);
kp1 = new(USnumberKeypad);
kp1.init(window);

PtSetParentWidget(window);
kp2 = new(FRalphaKeypad);
kp2.init(window);
kp2.SetPos(375,90);

PtRealizeWidget(window);
PtMainLoop();
```

6.1.2. Sample Keypad Definition Files

```
/*
 * Keypad definition file for English Numeric keypad
 */

SetGrid (30,30);
GridOffset (0, 0);
Font ("helv14b");
SetDim (9 * 30, 10 * 30 + .entryfield.dim.h + 4 * .entryfield.border_width);
SetButtons (0, 0, 3, 2, "7", "8", "9");
SetButtons (0, 2, 3, 2, "4", "5", "6");
SetButtons (0, 4, 3, 2, "1", "2", "3");
SetButtons (0, 6, 3, 2, ".", "0");
SignButton (6, 6, 3, 2, "+/-");
DelButton (6, 8, 3, 2, "<--");

Font ("helv10b");

CancelButton(0, 8, 2, 2, "Cancel");
ClearButton (2, 8, 2, 2, "Clear");
EnterButton (4, 8, 2, 2, "Accept");
```

```
/*
 * Keypad definition file for French Alpha keypad
 */

SetGrid (15,15);
GridOffset (0, 0);
SetDim (20 * 15, 11 * 15 + .entryfield.dim.h + 4 * .entryfield.border_width);
SetButtons (0, 0, 2, 2, "!", "@", "#", "$", "%", "^", "&", "*", "(", ")");
SetButtons (0, 2, 2, 2, "1", "2", "3", "4", "5", "6", "7", "8", "9", "0");
SetButtons (0, 4, 2, 2, "Q", "W", "E", "R", "T", "Y", "U", "I", "O", "P");
SetButtons (0, 6, 2, 2, "A", "S", "D", "F", "G", "H", "J", "K", "L", ".");
SetButtons (1, 8, 2, 2, "Z", "X", "C", "V", "B", "N", "M", "-", "_");

GridOffset (0, 0);
CancelButton (0, 11, 5, 2, "Cancelez");
ShiftButton (5, 11, 5, 2, "Shiftez");
DelButton (10, 11, 5, 2, "<--");
EnterButton (15, 11, 5, 2, "Acceptez");
SpaceButton (6, 13, 8, 2, "La Space");
```

6.1.3. The Keypad Class

```
/*
 * Create a Keypad based on a definition file.  We should be able to
 * handle multiple languages, not necessarily at runtime, but certainly
 * as part of the startup configuration.
 *
 * Todo:
 * 1) options for:
 *  - no titlebar
 *  - colors, so they are not hardcoded
 *  - block all windows?
 *
 * Revision 1.5  1999/05/17 18:36:46  manuel
 * Minor fixes: missing/unused locals; change local var names (max -> max_val).
 *
 * Revision 1.4  1999/04/23 19:02:57  manuel
 * Changed argument/variable name max to max_val to avoid conflist with
```

```
 *  the max function; also min -> min_val
 *
 * Revision 1.3  1999/02/01 16:36:47  sam
 * Keypad definitions are now loaded from the require_path, and the numeric
 * keypad has a "+/-" key.
 *
 * Revision 1.2  1999/01/11 22:42:10  sam
 * Switches now bring up a digital keypad.
 *
 * Revision 1.1  1999/01/09 02:13:51  sam
 * the slider templates now pop up a keypad to enter a number into
 *
 */

//require ("constants.g");
require_lisp ("Modal.lsp");
require_lisp ("PhotonWidgets.lsp");

/* ---------------------- KeypadButton Class ------------------------ */

class KeypadButton PtButton
{
}

method KeypadButton.constructor ()
{
  .fill_color = 0xaaaaff;
  .color = 0xffffff;
  .arm_fill = 1;
  .arm_color = 0x5555cc;
  .flags = cons (Pt_FOCUS_RENDER, nil);
  .SetDim (50,50);
  .highlight_roundness = 2;
}

/* ---------------------- KeypadText Class ------------------------ */

class KeypadText PtText
{
  entered_string;
  password_char;
}

method KeypadText.AddChars (str)
{
  if (! .password_char)
    {
      .text_string = string (.text_string, str);
      .cursor_position = strlen (.text_string);
    }
  else
    {
      .entered_string = string (.entered_string, str);
      .cursor_position = strlen (.text_string);
      .text_string = string(.text_string, substr("********************",
       0,strlen(str)));
    }
}

method KeypadText.constructor ()
{
  // .fill_color = 0xffffff;
  .highlight_roundness = 0;
  .text_font = "helv18";
  // .top_border_color = TOPBORDER;
  // .bot_border_color = BOTBORDER;
  .border_width = 4;
  .flags = Pt_ETCH_HIGHLIGHT | Pt_HIGHLIGHTED;
```

```
  .flags = cons (Pt_SET, nil);
  .text_flags = cons ( Pt_CURSOR_VISIBLE |
        Pt_EDITABLE |
        Pt_CHANGE_ACTIVATE, nil);

  .horizontal_alignment = Pt_RIGHT;
}

/* ------------------------ Keypad Class -------------------------- */

class Keypad
{
  window;
  entryfield;
  grid;
  gridoffset;
  shifted;
  blockwindows;
  blockallwindows;
  finished;
  buttonfont = "helv10b";

  isnumeric;
  max;
  min;
  range;

}

method Keypad.CreateButton (x, y, w, h, str)
{
  local  but;

  but = new (KeypadButton);
  but.SetPos (x * .grid.w + .gridoffset.x, y * .grid.h + .gridoffset.y);
  but.SetDim (w * .grid.w - 2 * but.border_width,
       h * .grid.h - 2 * but.border_width);
  but.text_string = str;
  but.text_font = .buttonfont;
  PtRealizeWidget (but);
  but;
}

method Keypad.Initialize()
{
  .window = new (PtWindow);
  .window.fill_color = 0xaadddd;
  // .window.flags = Pt_DELAY_REALIZE;
  .window.render_flags = cons(Ph_WM_RENDER_TITLE, nil);
  .entryfield = new (KeypadText);
  PtExtentWidget (.entryfield);
}

method Keypad.OutOfRangeMessage (msg_string)
{
  local val;

  val = .entryfield.text_string;

  .entryfield.text_string = msg_string;
  PtFlush();

  nanosleep(1,0);
  .entryfield.text_string = val;
}
```

```
/* ---------------------- Definition File Functions ---------------- */

/*
 * Functions callable through the definition file:
 * SetButton (x, y, w, h, str);
 * SetButtons (x, y, w, h, str ...);
 * SetExclButton (x, y, w, h, str);
 * SetExclButtons (x, y, w, h, str ...);
 * CancelButton (x, y, w, h, str);
 * ShiftButton (x, y, w, h, str);
 * SpaceButton (x, y, w, h, str);
 * DelButton (x, y, w, h, str);
 * EnterButton (x, y, w, h, str);
 * SetGrid (w, h);
 * GridOffset (x, y);
 * SetDim (w, h);
 * SetPos (w, h);
 */

method Keypad.SetDim (w, h)
{
  .window.SetDim (w, h);
  .entryfield.SetArea (0, 0, w - 4 * .entryfield.border_width, 24);
}

method Keypad.SetPos (x, y)
{
  .window.SetPos (x, y);
}

CLICK_EVENT = Pt_CB_ACTIVATE;

method Keypad.SetButton (x, y, w, h, str, exclusive ?= nil)
{
  local  but = .CreateButton (x, y, w, h, str);
  PtAttachCallback (but, CLICK_EVENT,
      '(@self).ButtonPress(@but, @exclusive));
  but;
}

method Keypad.SetExclButton (x, y, w, h, str)
{
  .SetButton(x, y, w, h, str, t);
}

method Keypad.SetButtons (x, y, w, h, strings...)
{
  local   str, but;

  with str in strings do
    {
      but = .SetButton (x, y, w, h, str);
      x += w;
    }
}

method Keypad.SetExclButtons (x, y, w, h, strings...)
{
  local   str, but;

  with str in strings do
    {
      but = .SetExclButton (x, y, w, h, str);
      x += w;
    }
}

method Keypad.CancelButton (x, y, w, h, str)
```

```
{
  local  but = .CreateButton (x, y, w, h, str);
  PtAttachCallback (but, CLICK_EVENT, '(@self).Cancel ());
  but;
}

method Keypad.NewlineButton (x, y, w, h, str)
{
  local  but = .CreateButton (x, y, w, h, str);
  PtAttachCallback (but, CLICK_EVENT, '(@self).Newline ());
  but;
}

method Keypad.ClearButton (x, y, w, h, str)
{
  local  but = .CreateButton (x, y, w, h, str);
  PtAttachCallback (but, CLICK_EVENT, '(@self).Clear ());
  but;
}

method Keypad.ShiftButton (x, y, w, h, str)
{
  local  but = .CreateButton (x, y, w, h, str);
  but.flags = Pt_TOGGLE;
  PtAttachCallback (but, CLICK_EVENT, '(@self).Shift ());
  but;
}

method Keypad.SpaceButton (x, y, w, h, str)
{
  local  but = .CreateButton (x, y, w, h, str);
  PtAttachCallback (but, CLICK_EVENT, '(@self).Space ());
  but;
}

method Keypad.DelButton (x, y, w, h, str)
{
  local  but = .CreateButton (x, y, w, h, str);
  PtAttachCallback (but, CLICK_EVENT, '(@self).Backspace ());
  // PtAttachCallback (but, REPEAT_EVENT, '(@self).Backspace ());
  but;
}

method Keypad.EnterButton (x, y, w, h, str)
{
  local  but = .CreateButton (x, y, w, h, str);
  PtAttachCallback (but, CLICK_EVENT, '(@self).Accept ());
  but;
}

method Keypad.SignButton (x, y, w, h, str)
{
  local  but = .CreateButton (x, y, w, h, str);
  PtAttachCallback (but, CLICK_EVENT, '(@self).ChangeSign ());
  but;
}

method Keypad.SetGrid (w, h)
{
  .grid = new (PhDim);
  .grid.w = w;
  .grid.h = h;
}

method Keypad.GridOffset (x, y)
{
  .gridoffset = new (PhPoint);
  .gridoffset.x = x;
```

```
  .gridoffset.y = y + .entryfield.dim.h + 4 * .entryfield.border_width;
}

method Keypad.Font (font)
{
  .buttonfont = font;
}

/* ----------------------------- Callbacks --------------------------- */

method Keypad.ButtonPress (button, exclusive ?= nil)
{
  local  str = button.text_string;
  if (.shifted)
    str = toupper (str);
  else
    str = tolower (str);

  if(exclusive) .Clear();

  .entryfield.AddChars (str);
}

method Keypad.Space ()
{
  .entryfield.AddChars (" ");
}

method Keypad.Newline ()
{
  .entryfield.AddChars ("\n");
}

method Keypad.Backspace ()
{
  local  str = .entryfield.text_string;
  .entryfield.text_string = substr (str, 0, strlen(str) - 1);

  if (.entryfield.password_char)
    {
      str = .entryfield.entered_string;
      .entryfield.entered_string = substr (str, 0, strlen(str) - 1);
    }

  .entryfield.AddChars ("");
}

method Keypad.Shift ()
{
  local  str = .entryfield.text_string;
  .shifted = !.shifted;
  if (.shifted)
    widget.flags = Pt_SET;
  else
    widget.flags = cons (Pt_SET,nil);
}

method Keypad.Accept ()
{
  local num;

  if (.isnumeric)
    {
      if (.max && .min)
 {
   num = number(.entryfield.text_string);

   if (num >= .min && num <= .max)
```

```
      .finished = #Accepted;
    else
      .OutOfRangeMessage(string("Valid Range: ",
           .min,"-",.max));

 }
    }
  else
    {
      if (.range)
 {
   if (strlen(.entryfield.text_string) > .range)
     {
       .OutOfRangeMessage(string("Too long: ",
     .range, " chars max"));
     }
   else
     {
       .finished = #Accepted;
     }
 }
      else
 {
   .finished = #Accepted;
 }
    }

}

method Keypad.Cancel ()
{
  .finished = #Cancelled;
}

method Keypad.Clear ()
{
  .entryfield.text_string = "";
  .entryfield.AddChars ("");
}

method Keypad.ChangeSign()
{
  local text = .entryfield.text_string;

  if(!text) text = "";

  .Clear();

  local lead = substr(text, 0, 1);

  if(lead == "-")
    {
      text = substr(text, 1, -1);
      if(!text) text = "";
    }
  else if(lead == "+")
    {
      text = substr(text, 1, -1);
      if(!text) text = "";
      text = string("-", text);
    }
  else
    {
      // assume positive
      text = string("-", text);
    }

  .entryfield.AddChars(text);
```

```
}

/* -------------- Initialization and Control -------------------- */

method Keypad.LoadDefinition (filename, parent, path ?= "")
{
  local  fptr, line, code;

  if (fptr = open(string(path, filename), "r", t))
    {
      PtSetParentWidget(parent);
      .Initialize ();
      PtSetParentWidget (.window);
      while ((line = read (fptr)) != _eof_)
 {
   /* Quote the method name */
   rplaca (line, list (#quote, car(line)));
   code = ‘call (@self, @@line);
   eval (code);
 }
      close (fptr);
    }
  else
    {
      error(string("Keypad failed to load ", path, filename, "!"));
    }
}

method Keypad.RemoveTitle ()
{
  .window.render_flags = cons(Ph_WM_RENDER_TITLE |
        Ph_WM_RENDER_RESIZE,nil);
}

method Keypad.ForceFront ( flag )
{
  if (flag)
    .window.state = Ph_WM_STATE_ISFRONT;
  else
    .window.state = Ph_WM_STATE_NORMAL;
}

method Keypad.Show ()
{
  .entryfield.text_string = "";
  PtRealizeWidget (.window);
  PtWidgetToFront(.window);
}

method Keypad.Hide ()
{
  PtUnrealizeWidget (.window);
}

method Keypad.BlockWindows (winlist)
{
  local return = .blockwindows;
  .blockwindows = winlist;
  return;
}

method Keypad.BlockAllWindows (flag)
{
  local return = .blockallwindows;
  .blockallwindows = flag;
  return;
}
```

```
method Keypad.SetPasswordChar ( character )
{
  .entryfield.password_char = character;
}

method Keypad.SetNumeric ( )
{
  .isnumeric = t;
}

method Keypad.ClearNumeric ()
{
  .isnumeric = nil;
}

method Keypad.SetRange (min_val, max_val)
{
  .min = min_val;
  .max = max_val;
}

method Keypad.ClearRange ()
{
  .min = .max = nil;
}

method Keypad.Input (defaultstr)
{
  local i, win;

  if(.blockallwindows)
    {
      with i in _photon_widgets_ do
 if(i != .window && class_of(i) == PtWindow)
   .blockwindows = cons(i, .blockwindows);
    }

  .Show();
  .entryfield.text_string = defaultstr;
  .entryfield.entered_string = "";
  .finished = nil;
  PtSetParentWidget (nil);
  with win in .blockwindows do
    win.flags = Pt_BLOCKED;
  modal (nil, #.finished);
  with win in .blockwindows do
    win.flags = cons (Pt_BLOCKED,nil);

  .Hide();

  if (.finished == #Accepted)
    .entryfield.text_string;
  else
    nil;
}

method Keypad.NumericInput (defaultval, min_val, max_val)
{
  // princ("call Keypad.NumericInput()\n");

  local ret_val;

  .SetNumeric();
  .SetRange(min_val,max_val);
  ret_val = .Input(defaultval ? string(defaultval) : "");
  .ClearRange();
  .ClearNumeric();
```

```
  // princ("result: ", ret_val, "\n");

  if (ret_val)
    {
      if(strlen(ret_val) > 0)
 number(ret_val);
      else
 nil;
    }
  else
    {
      nil;
    }
}

method Keypad.PasswordInput ()
{
  local ret_val, inval;

  .SetPasswordChar("*");
  .entryfield.entered_string = "";
  inval = .Input("");
  .SetPasswordChar(nil);

  if (inval)
    ret_val = .entryfield.entered_string;
  ret_val;
}
```

6.2. Handling Keyboard Events

This program creates a window and captures keystrokes, printing useful information to the console. Run it with the `-v` option to print a large amount of information. In QNX 4, run with:

```
[sh]$ phgamma kbd.g -v
```

or in QNX 6, with:

```
[sh]$ gamma kbd.g -v
```

To poll the keyboard in Photon, you actually set up an event handler as in this example, simply store all down keystrokes in a list, and remove them from the list when you receive the up keystroke. Create a timer (using the Gamma `every` function), and look at your list of currently pressed-down keys.

```
#!/usr/cogent/bin/phgamma

/* Bring in some general geometry methods for widgets */

require_lisp ("PhotonWidgets");

/* The mainline creates a blank window and attaches a raw callback to
   it.  This is how a window captures key events.  If you are looking
   for a particular key, the Pt_CB_HOTKEY callback is also useful. */

Verbose = nil;

function main ()
{
    PtInit (nil);
    w = new (PtWindow);
    w.SetDim (200,100);
    PtAttachCallback (w, Pt_CB_RAW, #cbHandleKeystroke(), Ph_EV_KEY);
```

```
    PtRealizeWidget (w);

    if (cadr(argv) == "-v")
     Verbose = t;
    PtMainLoop();
}

/* The event data is available as implicit local variables in the
   callback function:
   widget - the widget invoking this callback
   event_data   - the event data structure.
                Only one of the *_event unioned members of this
  structure is valid at any time.  Accessing the others
  may cause a crash.  The valid union member is
  determined by the type of callback being handled.  The
  member: rect is always valid.
   cbinfo        - the callback info structure.
                This is a union like event_data, where only one of
  the unioned members is valid at any one time.
  The members: event, reason, reason_subtype are always
  valid.
*/

/* The key symbols and flags are not built into the Gamma executable,
   but are readily available from /usr/include/photon/PkKeyDef.h. */

Pk_KF_Key_Down ::= 0x00000001;
Pk_KM_Shift ::= 0x00000001;
Pk_KM_Ctrl ::= 0x00000002;
Pk_KM_Alt ::= 0x00000004;

function cbHandleKeystroke ()
{
    if (Verbose)
    {
        princ ("Widget: ", class_name(widget), "\n");
        pretty_princ ("Evdata: ", event_data.key_event, "\n");
        princ ("CB Reason: ", cbinfo.reason, "\n");
        princ ("CB Subtype: ", cbinfo.reason_subtype, "\n");
        pretty_princ ("CB Event: ", cbinfo.event, "\n");
    }

    if ((event_data.key_event.key_flags & Pk_KF_Key_Down) != 0)
    {
        /* This is a down keystroke.  If the key_cap is above 256,
           then it is a key modifier, backspace, keypad, or other
           key that is not part of the standard ASCII key set. */
        if (Verbose || event_data.key_event.key_cap < 256)
        {
            princ ("\n");
            princ ("You pressed key cap ", event_data.key_event.key_cap,
                        " --> '");
            if (event_data.key_event.key_mods != 0)
            {
                if ((event_data.key_event.key_mods & Pk_KM_Ctrl) != 0)
                        princ ("ctrl-");
                if ((event_data.key_event.key_mods & Pk_KM_Alt) != 0)
                        princ ("alt-");
                if ((event_data.key_event.key_mods & Pk_KM_Shift) != 0)
                        princ ("shift-");
            }
            princ (char(event_data.key_event.key_cap), "'\n");
        }
    }

    if (Verbose)
            princ ("\n");
}
```

6.3. A CwGraph Rotating Cube

This code creates two graphs, one of which is a two-dimensional representation of a cube that appears to rotate in three-dimensional space. The graphs are created using the `CwGraph` widget, a contributed widget created at Cogent Real-Time Systems, Inc. This program uses a dynamic library, which requires that Gamma be called from it's own directory. For example, if Gamma is in your /usr/cogent/bin/ directory, you would start this program with the following command:

$ **`/usr/cogent/bin/phgamma graph.g 3d`**

The code starts here:

```
#!/usr/cogent/bin/phgamma

/*
 * This example shows two graphs made using CwGraph.
 * One graph is a 3-dimensional rotating cube.  You can
 * display this graph using the argument "3d" on the
 * command line.  The other graph shows 6 randomly-
 * generated traces of various colors.  No argument is
 * necessary on the command line to display this graph.
 */

/*
 * Load the required files, which include two dynamic
 * libraries that are not generally needed for Photon
 * widgets.
 */

require_lisp ("PhotonWidgets");
require_lisp("PhabTemplate.lsp");

use3d = nil;

/*
 * Used with the 3D cube example.
 */
xrot = 0;
yrot = 0;
zrot = 0;
pi = 3.141592653589;

/*
 * Used with the 2D graph example.
 */
DataSetSize = 100;
count = 0;
xlist1 = nil;

function main ()
{
 PtInit (nil);

 if (length(argv) > 1 && cadr(argv) == "3d")
  use3d = t;

 /* Set the redraw frequency. */

 if (use3d)
  every (0.1, #redraw());
 else
  every (0.1, #redraw2());
```

```
 MakeWindow ();

 PtRealizeWidget (w);
 PtMainLoop();
}

function MakeWindow ()
{
 w = new (PtWindow);
 gr = new (PtGroup);
 gr.SetDim (325,300);

 /* Anchor the group to the window.*/

 gr.group_flags = Pt_GROUP_STRETCH_HORIZONTAL |
                  Pt_GROUP_STRETCH_VERTICAL;
 gr.anchor_flags = Pt_LEFT_ANCHORED_LEFT |
                   Pt_RIGHT_ANCHORED_RIGHT |
     Pt_TOP_ANCHORED_TOP |
     Pt_BOTTOM_ANCHORED_BOTTOM;

 /* Assign graph variables.*/

 g = new (CwGraph);
 g.SetDim (325, 300);
 g.fill_color = 0x7777bb;
 g.graph_x_axis_color = 0x00ff00;
 g.graph_y_axis_color_left = 0x00ff00;
 g.graph_y_axis_color_right = 0x00ff00;
 g.graph_x_label_color = 0xffff00;
 g.graph_y_label_color_left = 0xffff00;
 g.graph_y_label_color_right = 0xffff00;

 if (!use3d)
 {
  g.graph_traces = 4;
  g.graph_xmin = 0;
  g.graph_xmax = DataSetSize;
  g.graph_x_label = "Test Trace";
  g.graph_y_label_left = "Y Axis";
  g.graph_y_label_right = "Alternate Axis";
  g.graph_x_font = "helv14bi";
  g.graph_y_font_left = "helv14b";
  g.graph_y_font_right = "helv14i";
  g.graph_flags = cons (Cw_GRAPH_Y_AXIS_LEFT |
          Cw_GRAPH_X_AXIS, -1);

  CwGraphSetYLimits (g, 0, 0, 1000);
  CwGraphSetYLimits (g, 1, 0, 1000);
  CwGraphSetYLimits (g, 2, 0, 1000);
  CwGraphSetYLimits (g, 3, 0, 1000);
  CwGraphSetTraceColor (g, 0, 0xff0000);
  CwGraphSetTraceColor (g, 1, 0x0000ff);
  CwGraphSetTraceColor (g, 2, 0x00ffff);
  CwGraphSetTraceColor (g, 3, 0x00ff00);
 }
 else
 {
  g.graph_traces = 6;
  g.graph_xmin = -2;
  g.graph_xmax = 2;
  g.graph_flags = cons (Cw_GRAPH_Y_AXIS_RIGHT |
          Cw_GRAPH_Y_MAJOR_TICK_RIGHT |
          Cw_GRAPH_Y_MINOR_TICK_RIGHT |
          Cw_GRAPH_Y_AXIS_TITLE_RIGHT |
          Cw_GRAPH_Y_AXIS_LABEL_RIGHT
          , nil);
```

```
  g.margin_width = 10;

  CwGraphSetYLimits (g, 0, -2, 2);
  CwGraphSetYLimits (g, 1, -2, 2);
  CwGraphSetYLimits (g, 2, -2, 2);
  CwGraphSetYLimits (g, 3, -2, 2);
  CwGraphSetYLimits (g, 4, -2, 2);
  CwGraphSetYLimits (g, 5, -2, 2);
  CwGraphSetTraceColor (g, 0, 0xff0000);
  CwGraphSetTraceColor (g, 1, 0xffff00);
  CwGraphSetTraceColor (g, 2, 0x00ff00);
  CwGraphSetTraceColor (g, 3, 0x00ff00);
  CwGraphSetTraceColor (g, 4, 0x00ff00);
  CwGraphSetTraceColor (g, 5, 0x00ff00);
 }
}

/*
 * Functions for the 3D example.
 */

function data (n, range, adder)
{
 local  i, l;

 for (i=0; i<n; i++)
 {
  l = cons (random() * range + adder, l);
 }
 l;
}

function xlist (start, n)
{
 local  i, l;
 for (i=start+n - 1; i>=start; i --)
  l = cons (i,l);
 l;
}

function newdata ()
{
 if (count > DataSetSize)
 {
  count = 0;
  CwGraphClearTrace (g,0);
  CwGraphClearTrace (g,1);
 }
 CwGraphAddXYPoints (g, 0, xlist(count,10), data(10,500,500));
 CwGraphAddXYPoints (g, 1, xlist(count,10), data(10,500,0));
 count += 10;
}

function mk3dpt (x, y, z)
{
 array (x, y, z);
}

function cube3d ()
{
 array (array (mk3dpt(-1,1,1), mk3dpt(-1,-1,1),
        mk3dpt(1,-1,1), mk3dpt(1,1,1),
        mk3dpt(-1,1,1)),
     array (mk3dpt(-1,1,-1), mk3dpt(1,1,-1),
     mk3dpt(1,-1,-1), mk3dpt(-1,-1,-1),
     mk3dpt(-1,1,-1)),
     array (mk3dpt(-1,1,1), mk3dpt(-1,1,-1)),
     array (mk3dpt(1,1,1), mk3dpt(1,1,-1)),
```

```
     array (mk3dpt(1,-1,1), mk3dpt(1,-1,-1)),
     array (mk3dpt(-1,-1,1), mk3dpt(-1,-1,-1))
     );
}

function identmatrix ()
{
 array (array (1,0,0,0),
        array(0,1,0,0),
        array (0,0,1,0),
        array (0,0,0,1));
}

function matxmat (mat1, mat2)
{
 local  i,j,res=array(array(0,0,0,0),
          array(0,0,0,0),
          array(0,0,0,0),
          array(0,0,0,0));

 for (i=0; i<4; i++)
 {
  for (j=0; j<4; j++)
  {
   res[i][j] = mat1[i][0] * mat2[0][j] +
        mat1[i][1] * mat2[1][j] +
        mat1[i][2] * mat2[2][j] +
        mat1[i][3] * mat2[3][j];
  }
 }
 res;
}

function vecxmat (vec, mat)
{
 local res = array (0,0,0,0);
 local i;

 for (i=0; i<4; i++)
 {
  res[i] = vec[0] * mat[0][i] +
           vec[1] * mat[1][i] +
    vec[2] * mat[2][i];
 }
 res;
}

function xrotmat (degrees)
{
 local  rad = degrees / 180 * pi, ret = identmatrix();
 ret[1][1] = ret[2][2] = cos (rad);
 ret[1][2] = sin (rad);
 ret[2][1] = -sin (rad);
 ret;
}

function yrotmat (degrees)
{
 local  rad = degrees / 180 * pi, ret = identmatrix();
 ret[0][0] = ret[2][2] = cos (rad);
 ret[0][2] = sin (rad);
 ret[2][0] = -sin (rad);
 ret;
}

function zrotmat (degrees)
{
 local  rad = degrees / 180 * pi, ret = identmatrix();
```

```
 ret[0][0] = ret[1][1] = cos (rad);
 ret[0][1] = sin (rad);
 ret[1][0] = -sin (rad);
 ret;
}

function redraw ()
{
 local  c = cube3d();
 local  mat = identmatrix();
 local  xr = xrotmat(xrot);
 local  yr = yrotmat(yrot);
 local  zr = zrotmat(zrot);
 local  traces = array();
 local  i=0, xpts, ypts, vec;

 xrot += 1;
 yrot += 1;
 zrot += 1;

 mat = matxmat (mat, xr);
 mat = matxmat (mat, yr);
 mat = matxmat (mat, zr);

 i=0;
 with trace in c do
 {
  xpts = nil;
  ypts = nil;
  with pt in trace do
  {
   vec = vecxmat (pt, mat);
   xpts = cons (vec[0], xpts);
   ypts = cons (vec[1], ypts);
  }
  traces[i] = cons (xpts,ypts);
  i++;
 }

 PtHold();
 for (i=0; i<6; i++)
 {
  CwGraphClearTrace (g,i);
  CwGraphAddXYPoints (g, i, car(traces[i]),
        cdr(traces[i]));
 }
 PtRelease();
}


/*
 * Function for the 2D example.
 */

function redraw2 ()
{
 local nanosleep = list;

 PtHold();
 {
  ylist1 = data (DataSetSize, 400, 600);
  ylist2 = data (DataSetSize, 400, 400);
  ylist3 = data (DataSetSize, 400, 200);
  ylist4 = data (DataSetSize, 400, 0);
  xlist1 = xlist (0, DataSetSize);
 }
 CwGraphClearTrace (g,0);
 CwGraphClearTrace (g,1);
```

```
 CwGraphClearTrace (g,2);
 CwGraphClearTrace (g,3);
 CwGraphAddXYPoints (g, 0, xlist1, ylist1);
 CwGraphAddXYPoints (g, 1, xlist1, ylist2);
 CwGraphAddXYPoints (g, 2, xlist1, ylist3);
 CwGraphAddXYPoints (g, 3, xlist1, ylist4);
 PtRelease();
}
```

6.4. A CwMatrix Spreadsheet

This code creates a simple spreadsheet, using the `CwMatrix` widget, a contributed widget created at Cogent Real-Time Systems, Inc. The program uses a dynamic library, which requires that Gamma be called from it's own directory. For example, if Gamma is in your /usr/cogent/bin/ directory, you would start this program with the following command:

$ **`/user/cogent/bin/phgamma matrix.g`**

There are only three spreadsheet functions implemented in this example, though more could be easily added. They are:

sum (range) produces the sum of a given range. The range is of the form A4:B7, and can be entered by typing **`=sum(`** followed by highlighting the range, followed by typing **`)`** into a spreadsheet cell, or by just typing in the range.

Example: **`=sum(A4:B7)`**

db (name) reflects the value of a Cascade DataHub point. This allows the example to display data from any live process that has a driver to the Cascade DataHub.

Example: **`=db(valve_1)`**

dbout (name,range) writes a value to the Cascade DataHub. The name is the Cascade DataHub point name, and the range indicates the cell whose value will be transmitted, which must be entered as a range from the cell to itself. This command will transmit its value whenever the sheet is recomputed.

Example: **`=dbout(valve_1_alarm, B5:B5)`**

The code starts here:

```
#!/usr/cogent/bin/phgamma

/*
 * This is a simple spreadsheet application written for QNX Photon(TM) with
 * the Gamma (TM) programming language.  It is not intended to be either
 * complete or correct, but merely to act as a demonstration of how to use
 * the CwMatrix widget.
 *
 * This code is provided free of charge or license by Cogent Real-Time
 * Systems Inc.  You are hereby given permission to use and modify this
 * code without restriction of any kind.
 */

/*
 * If we are running a Gamma executable that does not have the CwMatrix
 * widget defined, load the dynamic object libraries necessary to support
 * it.


if (undefined_p (CwMatrix))
{
  if (!dyna_add_lib ("/usr/cogent/lib/photon_s.dlb"))
    {
```

```
      princ ("Could not load dynamic library 'photon_s.dlb'\n");
      exit_program (1);
    }
  if (!dyna_add_lib ("/usr/cogent/lib/phwidgets.dlb"))
    {
      princ ("Could not load dynamic library 'phwidgets.dlb'\n");
      exit_program (1);
    }
}
 */
/*
 * Include the necessary utility libraries to enhance the Photon widget
 * set.  These files are found in /usr/cogent/lib
 */
require_lisp ("PhotonWidgets");
require ("PopupMenu");

/*
 * Declare some global variables.  The := syntax tells Gamma that only
 * the first attempt to set the variable will be honored.  This is useful
 * if you decide to re-read the source file while the application is
 * running.  It also allows you to set alternate values for these
 * variables through another program or from the command line prior to
 * loading this file.
 */

MENUHEIGHT := 34;
YOFFSET := 20;
XOFFSET := 20;
FormulaCells := make_array(0);

/*
 * Declare a sub-class of PtButton that has an extra resource called
 * sequence.  This sequence indicates the column or row number for which
 * this button is the header.  We will add DivButton widgets to dividers
 * for the row/col headers.
 */
class DivButton PtButton
{
  sequence;
}

/*
 * Declare a callback function for PtDivider to be called when the user
 * resizes a row.  We only need to resize the rows on either side of the
 * divider boundary, and then resize the last row in case the entire
 * divider has changed size.
 */
function cbDividerSetHeights (mat)
{
  local  left, right, sizes = widget.divider_sizes, n = length(sizes);

  left = cbinfo.divider.left;
  right = cbinfo.divider.right;
  mat.SetRowHeight (left.sequence, left.dim.h +
      left.border_width * 2, 0);
  mat.SetRowHeight (right.sequence, right.dim.h +
      right.border_width * 2, 0);
  mat.SetRowHeight (n - 1, sizes[n - 1].y - sizes[n - 1].x, 0);
  mat.FlushDamage ();
}

/*
 * Declare a callback function for PtDivider to be called when the user
 * resizes a column.
 */
function cbDividerSetWidths (mat)
{
```

```
  local  left, right, sizes = widget.divider_sizes, n = length(sizes);

  left = cbinfo.divider.left;
  right = cbinfo.divider.right;
  mat.SetColumnWidth (left.sequence, left.dim.w +
        left.border_width * 2, 0);
  mat.SetColumnWidth (right.sequence, right.dim.w +
        right.border_width * 2, 0);
  mat.SetColumnWidth (n - 1, sizes[n - 1].y - sizes[n - 1].x, 0);
  mat.FlushDamage ();
}

/*
 * Create the row label divider and buttons.  This is only called at
 * startup.
 */
function RowLabels (mat)
{
  local  i, divider, button;

  divider = new (PtDivider);
  divider.SetPos (0,YOFFSET + MENUHEIGHT);
  divider.SetDim (XOFFSET,mat.dim.h);
  divider.group_orientation = Pt_GROUP_VERTICAL;
  divider.divider_flags = cons (Pt_DIVIDER_RESIZE_BOTH, nil);
  PtAttachCallback (divider, Pt_CB_DIVIDER_DRAG,
      'cbDividerSetHeights (@mat));

  for (i=0; i<mat.matrix_rows; i++)
    {
      button = new (DivButton);
      button.sequence = i;
      button.text_string = string (i);
      button.border_width = 0;
      button.margin_height = 0;
      button.margin_width = 0;
      button.text_font = "helv10";
      button.border_width = 2;
      button.SetDim (XOFFSET - 2 * button.border_width,
       mat.RowHeight (i) - 2 * button.border_width);
    }
  PtRealizeWidget (divider);

  /* Since we cannot predict the size of the final button, we must
     adjust the final matrix row to match the button size. */
  mat.SetRowHeight (mat.matrix_rows - 1, button.dim.h +
      2 * button.border_width, 1);
}

/*
 * Create the column label divider and buttons.  This is only called at
 * startup.
 */
function ColumnLabels (mat)
{
  local  i, divider, button;

  divider = new (PtDivider);
  divider.SetPos (XOFFSET,MENUHEIGHT);
  divider.SetDim (mat.dim.w,YOFFSET);
  divider.group_orientation = Pt_GROUP_HORIZONTAL;
  divider.divider_flags = cons (Pt_DIVIDER_RESIZE_BOTH, nil);
  PtAttachCallback (divider, Pt_CB_DIVIDER_DRAG,
      'cbDividerSetWidths (@mat));

  for (i=0; i<mat.matrix_cols; i++)
    {
      button = new (DivButton);
```

```
      button.sequence = i;
      button.text_string = i < 26 ? char('A' + i) :
 string ("A", char('A' + i % 26));

      button.border_width = 0;
      button.margin_height = 0;
      button.margin_width = 0;
      button.text_font = "helv10";
      button.border_width = 2;
      button.SetDim (mat.ColumnWidth (i) - 2 * button.border_width,
       YOFFSET - 2 * button.border_width);

    }
  PtRealizeWidget (divider);
  /* Since we cannot predict the size of the final button, we must
     adjust the final matrix column to match the button size. */
  mat.SetColumnWidth (mat.matrix_cols - 1, button.dim.w +
        2 * button.border_width, 1);
}

/*
 * Compute the position of a menu button.  This is handy since we need
 * this position in order to determine where to put a pop_up menu.
 */
function MenuButtonPos (widget)
{
  local  pos;

  pos = PtGetAbsPosition (widget);
  pos.y += widget.dim.h + widget.border_width * 2;
  pos;
}

/*
 * Construct the template for the "Effects" menu, accessible through the
 * Effects button in the upper left corner.  This template does create an
 * actual PtMenu until you call its Instantiate(...) member function.
 */
function CreateEffectMenu (mat)
{
  local  emenu, textcolormenu, fillcolormenu;

  emenu = new (PopupMenu);
  emenu.AddItem ("Border On", nil, 'cbBorder(@mat,t), nil, nil);
  emenu.AddItem ("Border Off", nil, 'cbBorder(@mat,nil), nil, nil);
  emenu.AddItem ("-", nil, nil, nil, nil);
  emenu.AddItem ("Justify Left", nil, 'cbJustify(@mat,Cw_CELL_LEFT),
   nil, nil);
  emenu.AddItem ("Justify Right", nil, 'cbJustify(@mat,Cw_CELL_RIGHT),
   nil, nil);
  emenu.AddItem ("Justify Center", nil, 'cbJustify(@mat,Cw_CELL_CENTER),
   nil, nil);
  emenu.AddItem ("-", nil, nil, nil, nil);
  emenu.AddItem ("Invert", nil, 'cbInvert(@mat, 1), nil, nil);
  emenu.AddItem ("Un-Invert", nil, 'cbInvert(@mat, 0), nil, nil);
  emenu.AddItem ("-", nil, nil, nil, nil);

  fillcolormenu = new (PopupMenu);
  fillcolormenu.AddItem ("red", nil,
    'cbFillColor(@mat,0xff0000), nil, nil);
  fillcolormenu.AddItem ("green", nil,
    'cbFillColor(@mat,0x00ff00), nil, nil);
  fillcolormenu.AddItem ("blue", nil,
    'cbFillColor(@mat,0x0000ff), nil, nil);
  fillcolormenu.AddItem ("white", nil,
    'cbFillColor(@mat,0xffffff), nil, nil);
  fillcolormenu.AddItem ("black", nil,
    'cbFillColor(@mat,0x000000), nil, nil);
```

```
  textcolormenu = new (PopupMenu);
  textcolormenu.AddItem ("red", nil,
    'cbTextColor(@mat,0xff0000), nil, nil);
  textcolormenu.AddItem ("green", nil,
    'cbTextColor(@mat,0x00ff00), nil, nil);
  textcolormenu.AddItem ("blue", nil,
    'cbTextColor(@mat,0x0000ff), nil, nil);
  textcolormenu.AddItem ("white", nil,
    'cbTextColor(@mat,0xffffff), nil, nil);
  textcolormenu.AddItem ("black", nil,
    'cbTextColor(@mat,0x000000), nil, nil);

  emenu.AddItem ("Text Color", nil, textcolormenu, nil, nil);
  emenu.AddItem ("Fill Color", nil, fillcolormenu, nil, nil);

  emenu;
}

/*
 * Various functions to be called when the menu buttons are selected.
 */

/* Turn on/off borders around the currently selected region */
function cbBorder (mat, on)
{
  local  flags = on ? Cw_CELL_ALL_BORDERS_THICK : 0;

  mat.OutlineRange (mat.matrix_range, flags, Cw_CELL_ALL_BORDERS_THICK);
  mat.FlushDamage();
}

/* Justify left/right/center in the currently selected region */
function cbJustify (mat, flag)
{
  mat.JustifyRange (mat.matrix_range, flag);
  mat.FlushDamage();
}

/* Invert text/background colors in the currently selected region */
function cbInvert (mat, yesno)
{
  mat.InvertRange (mat.matrix_range, yesno);
  mat.FlushDamage();
}

/* Set the background color in the currently selected region */
function cbFillColor (mat, color)
{
  local  row, col, range, cell;

  range=mat.matrix_range;
  for (row=range.ul.y; row<=range.lr.y; row++)
    {
      for (col=range.ul.x; col<=range.lr.x; col++)
 {
   if (cell = mat.Cell (row, col))
     cell.fill_color = color;
 }
    }
}

/* Set the text color in the currently selected region */
function cbTextColor (mat, color)
{
  local  row, col, range, cell;

  range=mat.matrix_range;
```

```
  for (row=range.ul.y; row<=range.lr.y; row++)
    {
      for (col=range.ul.x; col<=range.lr.x; col++)
 {
   if (cell = mat.Cell (row, col))
     cell.text_color = color;
 }
    }
}

/*
 * This is a callback function that is called when the user begins to
 * edit the contents of a cell.  This gives us an opportunity to insert
 * a different text string from the one shown for editing purposes.  In
 * this case, if the cell has a formula, edit the formula instead of the
 * visible text.
 */
function cbBeginEdit ()
{
  if (cbinfo.matrix.cell.formula)
    cbinfo.matrix.text_string = string ("=",cbinfo.matrix.cell.formula);
}

/*
 * This is a callback function that is called when the text in a cell
 * has changed.  We check to see whether this is a formula, and if so
 * attempt to parse it and evaluate it.
 * In any case, we must re-compute the contents of the sheet in case
 * this edit has affected something elsewhere.
 */
function cbTextChange ()
{
  local  answer, str, _parser_throws_error_ = t;

  str = cbinfo.matrix.text_string;
  if (str[0] == '=')
    {
      str = substr(str,1,-1);
      SetFormula (cbinfo.matrix.cell, str);
      EvaluateCell (widget, cbinfo.matrix.cell.row, cbinfo.matrix.cell.col);
    }
  else
    SetFormula (cbinfo.matrix.cell, nil);
  Recompute (widget);
}

/*
 * This is a bsearch comparison function that orders the cells in row/column
 * order.  If we wanted to evaluate in column/row order, we would modify
 * this function to suit.
 */
function CmpCellPos (!c1, !c2)
{
  if (c1.row < c2.row)
    -1;
  else if (c1.row == c2.row)
    c1.col - c2.col;
  else
    1;
}

/*
 * Add a cell containing a formula to the global formula array.  We do this
 * so that when we recompute the sheet, we only have to visit those cells
 * that are known to have a formula.  Since a sheet is usually sparse, this
 * can save a lot of time.
 */
function AddFormulaCell (cell)
```

```
{
  local  found;

  found = bsearch (FormulaCells, cell, CmpCellPos);
  if (undefined_p (car(found)))
    insert (FormulaCells, cdr(found), cell);
}

/*
 * Remove a cell from the global formulas array.
 */
function RemoveFormulaCell (cell)
{
  local  found;

  found = bsearch (FormulaCells, cell, CmpCellPos);
  if (!undefined_p(car(found)))
    delete (FormulaCells, cdr(found));
}

/*
 * Set the formula for a cell.  By offering this single entry point for
 * modifying the formula, we can ensure that we know which cells to
 * look at when we recompute the sheet.
 */
function SetFormula (cell, str)
{
  if (!str || str == "")
    RemoveFormulaCell (cell);
  else
    AddFormulaCell (cell);
  cell.formula = str;
}

/*
 * Evaluate the formula of a cell, and set the cell's visible text string
 * to the result.  We attempt this once using the exact formula text.  If
 * it fails, we append a semicolon to the formula and try again.  If there
 * is an error, we do something cheesy to get rid of some of the error
 * message, since it could be much too long for the cell on-screen.
 */
function EvaluateCell (mat, row, col)
{
  local  answer, str, _parser_throws_error_ = t;

  if (cell = mat.ExistingCell (row, col))
    {
      str = cell.formula;
      if (str && str != "")
 {
   try
     {
       answer = eval_string (str,t);
     }
   catch
     {
       try
  {
    answer = eval_string (string (str, ";"),t);
  }
       catch
  {
    answer = string_split(_last_error_, ":", 3);
    if (length(answer) == 4)
      answer = car(reverse(answer));
    else
      answer = _last_error_;
  }
```

```
    }
   cell.text_string = string (answer);
 }
    }
}

/*
 * This is an exhaustive recomputation function that looks to see if a
 * cell has any data, and then attempts to evaluate it.  It is not very
 * efficient, so we don't use it.  It is just here for demonstration.
 */
function Recompute (mat)
{
  local   row, col, cell;

  for (row = 0; row < mat.matrix_rows; row ++)
    {
      for (col = 0; col < mat.matrix_cols; col ++)
 {
   if (cell = mat.ExistingCell (row, col))
     {
       EvaluateCell (mat, row, col);
     }
 }
    }
}

/*
 * This is the real recomputation function.  We keep track of which cells
 * contain a formula so we only have to deal with those.  It cannot help
 * but be faster.  The FormulaCells are sorted in row/col order, so we
 * do not need to do anything fancy to get the evaluation order correct.
 */
function Recompute (mat)
{
  with cell in FormulaCells do
    {
      EvaluateCell (mat, cell.row, cell.col);
    }
}

/*
 * A little Gamma cheat.  : (colon) is a function taking two arguments.  We
 * override it here, and just have it generate a rectangle from its args.
 * Now we can express ranges as A2:B5 without breaking the Gamma syntax.
 */
function \: (!x,!y)
{
  local  rect = new (PhRect);
  SpreadsheetToPoint (string(x), rect.ul);
  SpreadsheetToPoint (string(y), rect.lr);
  rect;
}

/*
 * Convert a number to an alpha number for the column designation.
 */
function ConvertToAlpha (num)
{
  num < 26 ? char('A' + num) :
    string ("A", char('A' + num % 26));
}

/*
 * Convert a range rectangle into a A2:B4 style designation.
 */
function cbRangeConvert ()
{
```

```
  local   str, rect = cbinfo.matrix.cur_range;

  str = string (ConvertToAlpha(rect.ul.x), rect.ul.y, ":",
  ConvertToAlpha(rect.lr.x), rect.lr.y);
  cbinfo.matrix.text_string = str;
}

/*
 * The mainline.  Here we set up interprocess communication and build the
 * main screen.
 */
function main ()
{
  local  w, mat, menubar, mbutton, emenu;

  init_ipc ("matrix","matrix");
  PtInit (nil);

  w = new (PtWindow);
  w.SetDim (400, 300);
  w.resize_flags = cons (Pt_RESIZE_X_INITIAL | Pt_RESIZE_Y_INITIAL, -1);
  menubar = new (PtMenuBar);
  menubar.SetDim (menubar.dim.w, MENUHEIGHT - 2 * menubar.border_width);
  mbutton = new (PtMenuButton);
  mbutton.text_string = "Effects";

  PtRealizeWidget (w);

  PtSetParentWidget (w);
  mat = new (CwMatrix);
  mat.matrix_rows = 10;
  mat.matrix_cols = 10;
  mat.SetArea (XOFFSET, YOFFSET + MENUHEIGHT, 400 - XOFFSET,
        300 - (YOFFSET + MENUHEIGHT));
  mat.anchor_flags = cons (Pt_LEFT_ANCHORED_LEFT | Pt_RIGHT_ANCHORED_RIGHT |
      Pt_TOP_ANCHORED_TOP | Pt_BOTTOM_ANCHORED_BOTTOM,
      -1);
  PtAttachCallback (mat, Cw_CB_MATRIX_BEGIN_EDIT, `cbBeginEdit());
  PtAttachCallback (mat, Cw_CB_MATRIX_TEXT_CHANGE, `cbTextChange());
  PtAttachCallback (mat, Cw_CB_MATRIX_RANGE_CONVERT, `cbRangeConvert());

  PtRealizeWidget (mat);
  PtSetParentWidget (w);
  RowLabels(mat);
  PtSetParentWidget (w);
  ColumnLabels(mat);

  emenu = CreateEffectMenu(mat);
  PtAttachCallback (mbutton, Pt_CB_ARM,
      `(@emenu).Instantiate (@mbutton,
        MenuButtonPos(@mbutton)));

  PtMainLoop ();
}

/* ---------------------- Spreadsheet functions ---------------- */

/*
 * This section contains functions that are callable from a spreadsheet
 * cell.  We need to specify these specially, as they must expect to
 * take a range as input.  Unfortunately, this means that we cannot
 * simply use the functions built into Gamma.
 */

function SpreadsheetToPoint (str, point)
{
  local  i, row, col, colstr;
```

```
  for (i=0; str[i] >= 'A' && str[i] <= 'Z'; i++);
  colstr = substr(str,0,i);
  row = number (substr(str,i,-1));
  if (i==1)
    col = str[0] - 'A';
  else
    col = (str[0] - 'A') * 26 + str[1] - 'A';
  point.x = col;
  point.y = row;
  point;
}

/*
 * Functions usable in the spreadsheet.  There are global variables
 * cell and mat defined during these function.
 *
 * A range is a PhRect representing the upper left and lower right
 * corners of the range.
 *
 * In a spreadsheet cell, you would see:  =sum(A5:B6)
 */

function sum (range)
{
  local  rect = range, row, col, cell, result=0;

  for (row = rect.ul.y; row <= rect.lr.y; row++)
    {
      for (col = rect.ul.x; col <= rect.lr.x; col++)
 {
   if (cell = mat.ExistingCell(row,col))
     {
       result += number(cell.text_string);
     }
 }
    }
  result;
}

/*
 * Exception function that updates a cell when a &datahub; point
 * change occurs, and then recomputes the spreadsheet.
 */
function exCellException(cell)
{
  cell.text_string = string (value);
  Recompute(mat);
}

/*
 * A spreadsheet function that causes a cell to be attached to a
 * &datahub; point.
 * In the cell you would see:  =db(tank_level)
 * Notice that the point name is unevaluated, so you do not need to
 * use the # or ' syntax to protect the point name.
 */
function db (!name)
{
  local  sym = symbol(name);
  if (!getprop (sym,#registered))
    {
      setprop (sym, #registered, t);
      register_exception (sym, 'exCellException(@cell));
    }
  eval(sym);
}

/*
```

```
 * A spreadsheet function that writes a value to the &datahub;.
 * This function transmits a value from the top-left corner of the given
 * range to the named point every time it is evaluated.  This will not
 * transmit the value of the current cell, since this would be a bit of
 * a chicken-and-egg problem.  The result in the cell will be the point
 * name on success, or an error message on failure.
 */
function dbout (!name,range)
{
  local  sym = symbol(name);
  local  rect = range, outcell;

  if (outcell = mat.ExistingCell (rect.ul.y, rect.ul.x))
    {
      write_point (sym, number(outcell.text_string));
      name;
    }
  else
    "No value";
}
```

Appendix A. GNU General Public License

GNU General Public License

Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software - to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps:

1. copyright the software, and
2. offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

Section 0

This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program",

below, refers to any such program or work, and a “work based on the Program” means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term “modification”.) Each licensee is addressed as “you”.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

Section 1

You may copy and distribute verbatim copies of the Program’s source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

Section 2

You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: If the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

Section 3

You may copy and distribute the Program (or a work based on it, under Section 2 in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

Section 4

You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

Section 5

You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

Section 6

Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

Section 7

If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or

otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

Section 8

If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

Section 9

The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

Section 10

If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY Section 11

BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

Section 12

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

<one line to give the program’s name and a brief idea of what it does.> Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type “show w”. This is free software, and you are welcome to redistribute it under certain conditions; type “show c” for details.

The hypothetical commands “show w” and “show c” should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than “show w” and “show c”; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program “Gnomovision” (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989 Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

Appendix B. GNU Lesser General Public License

GNU Lesser General Public License

Version 2.1, February 1999

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method:

1. we copyright the library, and
2. we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author’s reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the *Lesser* General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

Section 0

This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

Section 1

You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

Section 2

You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a. The modified work must itself be a software library.

b. You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c. You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d. If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

   (For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

Section 3

You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

Section 4

You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

Section 5

A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a “work that uses the Library”. Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a “work that uses the Library” with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a “work that uses the library”. The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a “work that uses the Library” uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

Section 6

As an exception to the Sections above, you may also combine or link a “work that uses the Library” with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer’s own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work

during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a. Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable “work that uses the Library”, as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b. Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user’s computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c. Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d. If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e. Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the “work that uses the Library” must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

Section 7

You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a. Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b. Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

Section 8

You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who

have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

Section 9

You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

Section 10

Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

Section 11

If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

Section 12

If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

Section 13

The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

Section 14

If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY Section 15

BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

Section 16

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

<one line to give the library’s name and a brief idea of what it does.> Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library ’Frob’ (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990 Ty Coon, President of Vice

That’s all there is to it!

Colophon

This book was produced by Cogent Real-Time Systems, Inc. from a single-source group of SGML files. Gnu Emacs was used to edit the SGML files. The DocBook DTD and related DSSSL stylesheets were used to transform the SGML source into HTML, PDF, and QNX Helpviewer output formats. This processing was accomplished with the help of OpenJade, JadeTeX, Tex, and various scripts and makefiles. Details of the process are described in our book: Preparing Cogent Documentation, which is published on-line at
`http://developers.cogentrts.com/cogent/prepdoc/book1.html`.

Text written by Bob McIlvride.