HEXAGON
GEOSPATIAL

IMAGINE DEVELOPERS' TOOLKIT™

White Paper
December 23, 2014

Product Overview

The IMAGINE Developers' Toolkit™ is a set of libraries and documentation for ERDAS IMAGINE® users who wish to modify the commercial version of the software, or develop entirely new applications to extend the capabilities of the software, to meet their specific project needs. The IMAGINE Developers' Toolkit includes a set of "C/C++" language API's that are intended for the experienced "C/C++" programmer. Those customers who have purchased software maintenance for the IMAGINE Developers' Toolkit will also be granted access to the Intergraph® Toolkit Developers' Network. This network is an online, interactive tool used to present the most up-to-date documentation and examples for the Developers' Toolkit API. This document describes what sort of modifications and new applications may be developed using the IMAGINE Developers' Toolkit.

ERDAS IMAGINE is a broad suite of software products designed to address the needs of a wide audience. Although the user interface is designed to make workflows easy for a variety of skill and experience levels, some organizations need to customize the software in order to streamline a particular production workflow. The basic customization capabilities of ERDAS IMAGINE are particularly well suited for modifying the easy-to-use graphical interface, while the IMAGINE Developers' Toolkit is designed primarily for users needing to tailor the software beyond this level.

What type of customization/modifications do you need to make? This is the most important question in determining which ERDAS IMAGINE tools you will need to use. ERDAS IMAGINE offers two broad categories of customization capabilities, each achievable with different sets of tools:

- Customizing
- Extending

Customizing

ERDAS IMAGINE provides a range of customization capabilities — from simple preferences to complete changes to the user interface. This helps organizations that, for example, wish to change the language of the interface or need to simplify it for dedicated applications, such as photo-interpretation.

Preferences

The simplest and most straightforward means of customizing is provided by the Preference Editor. The Preference Editor manages a database of preference values that are used throughout the product to modify its behavior. A preference is a value that defines a user's choice for some optional aspect of the software. For example, ERDAS IMAGINE can display "bubble help" whenever the cursor lies on top of a user interface element for some period of time. The user may define the length of time before bubble help appears, as well as choose to have it never appear by setting preferences. Preferences may be saved in either a local or a global sense. A local change affects only the current user by saving the settings in his/her home directory. A global change is saved in the ERDAS IMAGINE installation directory and affects all users (unless overridden by a local change). The Preference Editor is provided with every ERDAS IMAGINE license.

With the number and variety of imagery formats always on the rise, the data importer and exporter is the most common type of application added to ERDAS IMAGINE. The IMAGINE Developers' Toolkit contains functions which provide I/O for the ERDAS IMAGINE file format, as well as functions that make it easier to create an importer/exporter that operates like those that ship with ERDAS IMAGINE

EML Script Changes

The ERDAS IMAGINE graphical user interface was developed with the ERDAS Macro Language (EML), which comes with every license of the software. This language is a scripting language that can be used to define the structure and content of the user interface, as well as provide some fundamental procedural scripting capabilities. Each script is interpreted at application startup and converted to instructions for the native windowing system (i.e., EML user interface constructs are converted to Motif under UNIX and Win32 under Windows). These files are included in a virtual scripts directory when the associated application is built. Instead of being "compiled", they are just copied to the IMAGINE_HOME directory for runtime. Each script is an ASCII file, which may be edited to change its contents. For example, the titles of all menus, buttons, and other user controls may be changed to another language. In addition, menu items may be added or deleted. In many cases (though not all), commands may be added to or deleted from existing dialogs.

Extending

In addition to customizing the existing applications, some organizations may want to add new capabilities to the software, such as adding to the existing scripts, writing new Spatial Modeling Language (SML) scripts, or developing entirely new applications using the IMAGINE Developers' Toolkit.

EML Script Additions

Since EML provides a procedural-scripting environment in addition to the user interface definition, it is possible to create new capabilities by combining existing ERDAS IMAGINE commands under new menu items or new buttons.

SML Additions

New applications can be built using SML, which is a component of both IMAGINE Advantage® and IMAGINE Professional® licenses. The Image Interpreter in ERDAS IMAGINE is built primarily from SML scripts with an EML interface. Once you have developed a new algorithm using the graphical Modell Maker environment, you can generate an SML script that can be combined with a custom-developed user interface. The new application may be plugged into the existing ERDAS IMAGINE menu structure so that it functions like any other part of the system. Example spatial models can be downloaded from our Intergraph Toolkit Developers' Network.

Custom Applications

Extending ERDAS IMAGINE goes beyond changing interfaces and adding new image processing applications, and consequently generally requires the use of the IMAGINE Developers' Toolkit. One of the keys to making the system useful is the ability to use any data available, or to integrate entirely new types of image processing.

- Importers/Exporters

Custom DLLs

A major feature of the ERDAS IMAGINE architecture is the use of DLLs to provide for custom extensions. A DLL is a Dynamically Loadable Library, which is a piece of code to be located and used at run time by an application. ERDAS IMAGINE uses this to create plug-in sockets in a variety of areas, such as imagery access, coordinate conversion, coordinate projection, and font access. Without modification to existing applications, a Raster Format DLL may be written and added to the system, which allows all ERDAS IMAGINE applications to access data stored in previously unsupported file formats directly and without file conversion.

Product Definition and Requirements

The IMAGINE Developers' Toolkit is the collection of libraries and associated headers and documentation, used by the Intergraph development staff to create Intergraph products. The libraries are organized into several large functional groups. The core is the base library, called erasterlib (ERDAS Raster Library), which contains all of the core functionality needed to create/read/write geospatial imagery. Layered on top of this is elib (ERDAS Library), which contains all of the non-GUI components of application creation, such as session management, annotation I/O, and a number of utility packages. Layered on top of the elib library is the user interface library, called emllib (ERDAS Macro Language Library), which contains all of the functions for creating a portable GUI and interfacing with EML.

The Intergraph Toolkit Developers' Network (http://developer.lggi.com) is an interactive online network of IMAGINE Toolkit Programmers. All content in this environment is shared by Intergraph Toolkit Customers and Intergraph employees which makes this a true network of programmers. This method for presenting the Toolkit documentation to every programmer using it will insure that all content in the network be as up-to-date as possible. There are several parts to this web environment:

1. The most important feature is the Toolkit Library which contains all supported Toolkit packages, data types, and functions. All entries are categorized for easy searching and every item is guaranteed to be complete.
2. Developers' FAQ section is a list of commonly asked questions with answers provided by Intergraph employees.
3. Each item in the Toolkit Library Database is dynamically linked to every example, This means that when selecting a particular function or data type, a list of example program that contain that item will be returned. There is also a categorical search for Visual C/C++, EML, SML, and spatial models.
4. Each user will be able to download the complete project from the Examples Database. There is also a categorical search in place to help when downloading several examples at a time.
5. We have developed a peer-to-peer coding forum so that customers are able to ask questions on any developer-related topic.
6. RSS feed is available to those customers who are interested in knowing exactly when updates to the network are made. The feed will include functions, data types, examples, and information about the Developers' Network.

Users of the IMAGINE Developers' Toolkit must be experienced C programmers. The IMAGINE Developers' Toolkit is available as an add-on module for all ERDAS IMAGINE users. Support is offered as a separate package designed specifically to meet the needs of developers. Access to download content from the Intergraph Toolkit Developers' Network is only available to those customers with a current software maintenance contract.

Problem Solving With IMAGINE Developers' Toolkit

Example 1: A New Importer

The manufacturer of the Daedalus airborne scanner needed to make their data available to potential customers who also used ERDAS IMAGINE. The data is on 8mm tape and needs to be imported into an ERDAS IMAGINE compatible file format. The importer they developed uses the ERDAS IMAGINE tape functions to access the tape device, which provides a portable means of interfacing to the tape on Windows and UNIX. As the data is read in, it is written to an ERDAS IMAGINE .img file using the Image I/O functions in elib. The geographic information is also used to create input to ERDAS IMAGINE rectification tools to provide a semi-automated process of geocorrection.

Example 2: A New Sensor Model

A military contractor, developing a system for photointerpretation of satellite data, had a sensor that did not use a geometry similar to any of the typical frame cameras or push broom sensors, and they wanted to use the imagery without resampling in the ERDAS IMAGINE application. To solve this problem, they created a plug-in sensor model for use with ERDAS IMAGINE's geometric modeling. The sensor model describes the geometry of their sensor (e.g. it tells how to convert a pixel coordinate – row and column – into a ground coordinate) and how to perform the inverse operation. The model was implemented as an ERDAS IMAGINE Geometric Model DLL. It allowed ERDAS IMAGINE to then be used to read ground base coordinates from the image, as well as orthocorrect the image with the existing ERDAS IMAGINE tools.

Example 3: ATCOR 2 - A New Application

Dr. Rudolph Richter of DLR in Germany is a leading specialist in the area of atmospheric correction. His algorithms, known as ATCOR, have been widely used to remove the effects of the atmosphere from satellite images. The ERDAS IMAGINE distributor in Germany, GEOSYSTEMS GmbH, collaborated with Dr. Richter and used the IMAGINE Developers' Toolkit to create the ATCOR 2 module for use with ERDAS IMAGINE. This software is similar to other ERDAS IMAGINE add-on modules, and when installed, it adds a new icon to the top-level menu bar. When activated, it displays a detailed user interface that interacts with the ERDAS IMAGINE Viewer and displays data using the ERDAS IMAGINE charting tool. These tools are used to fine-tune the atmospheric model before correction takes place.

Example 4: TMWS - A Photointerpreter Workstation

One of the earliest applications of the IMAGINE Developers' Toolkit was the development of the Target Materials Workstation, or TMWS. TMWS was developed to provide an easy-to-use workstation for military photointerpreters.

The interpreter's task is to look at an image, mark up any important features, and then print the results. They must be able to sharpen, contrast stretch, and perhaps rotate and scale the image. Then, annotation tools are used to place text, lines, and other well-defined graphics on the image. This marked-up image is then inserted into one of several standard map templates and printed for dissemination. In addition, there is usually some central control of tasking.

TMWS was made by customizing ERDAS IMAGINE. The menus and tool palettes were simplified and tailored for the task at hand. In addition, special top-level icons were created to lead the operator through the process. Coordination of the workflow and data management was achieved by working with an already existing central Sybase database. This was done by using Sybase libraries in conjunction with the IMAGINE Developers' Toolkit to create a module which could obtain information about the images to be analyzed, and then record the name and location of the final products.

Key Features of the IMAGINE Developers' Toolkit

- ERDAS IMAGINE object manipulation routines
- Application environment routines
- Low-level file I/O and system access routines
- Abstract object manipulation routines
- Unicode storage classes
- EML GUI access routines extensive context-sensitive on-line help
- Peer-to-peer forum available through the Developers' Network
- Example programs
- LPS block file creation routines
- 2D Visualization routines

Technical Specifications of the IMAGINE Developers' Toolkit

Architecture

Routines

The IMAGINE Developers' Toolkit is organized as a number of packages which create and support a type of data object. The package usually defines a few data types, which are then manipulated by the functions. These packages are described below in groups of routines.

DLL Classes

Some of the packages are built upon a DLL mechanism, which allows that package to be extended dynamically. For example, the eimg routines for reading and writing imagery from files are based upon the Raster Formats DLL Class. Any application written using the eimg package will be able to read data from any of the supported file formats. (The packages based upon an extensible DLL are indicated by the name of the DLL class in brackets, for example [Raster Formats DLL])

ERDAS IMAGINE Object Manipulation Routines

Annotation

The Annotation (eant) package provides functions for creating, reading, writing, rendering, and editing ERDAS IMAGINE annotation elements and their styles. Annotation elements consist of polylines, polygons, ellipses, elliptical arcs, points, text, and groups.

Area of Interest

The Area of Interest (eaoi) package provides basic tools for area of interest analysis. It includes AOI structure operations (create, delete, copy, etc.), AOI input and output operations, and AOI rendering operations.

Color Library

The Color Library (eclb) package provides support for color libraries, RGB/IHS conversions, and level slicing.

Unit Conversion

The Unit Conversion (ecvt) package provides routines for converting between units of distance, area, time, volume, angle, etc. The units supported are defined in a database ("units.dat"), which defines the relation between a "standard" unit and others. For example, the standard distance is in meters, all other distance units are defined as multiples of meters.

Descriptor Table Access

The Descriptor Table Access (edsc) package provides basic tools for manipulating descriptor tables. A descriptor table is the part of an ERDAS IMAGINE .img file that stores ancillary information about a particular data value or set of data values in a raster layer. The information for each data value, or set of data values, might include a histogram value, color table look-up values, a class name, total area, etc. The edsc functions allow the programmer to easily create in-memory data structures for descriptor tables and to easily pass descriptor table information between memory and disk without having to deal with the lower level ehfa, emif, and efio packages. The functions allow the programmer to do the following:

- Create, open, and close a descriptor table
- Create, copy, and delete a bin function (the part of the descriptor table that describes how data values in the raster layer are mapped to rows in the descriptor table)
- Create, open, read, write, and close a descriptor table column (the ancillary information).

Digital Signal Processing

The Digital Signal Processing (edsp) package provides functions that perform one- and two-dimensional forward and inverse Fast Fourier Transformations (FFTs) on complex or real data. Most functions in this routine have been specially designed for efficient transformation of real valued sequences and arrays. These sequences have certain symmetries that reduce both storage requirements and the number of computations. The length of each sequence must be a power of two.

Graphics Support

The Graphics Support (eevg) package provides functions for rendering vector-type data (annotation, vectors) to output devices. It handles the rendering of line styles, polygon fill styles, and smooth outline scalable text. Floating-Point Graphics Arithmetic The Floating-Point Graphics Arithmetic (efga) package provides routines for performing commonly used vector, matrix, point, and polynomial (including affine) transformation functions.

Floating-Point Graphics Arthmetic

The Floating-Point Graphics Arithmetic (efga) package provides routines for performing commonly used vector, matrix, point, and polynomial (including affine) transformation functions.

File Node Parse

The File Node Parse (efnp) package breaks up strings containing file names and ehfa node names and ranges of ehfa node names into their component parts. For example, it can be used to separate the path "D:/data/lanier.img(:Layer_1)" into the components "D:\data\," "lanier.img" and "(:Layer_1)."

General Data Arithmetic

The General Data Arithmetic (egda) package provides basic tools to perform operations on various data objects based on the Egda_BaseData structure. Objects based on this data structure are prevalent in the IMAGINE Developers' Toolkit libraries (e.g., the eimg_PixelRect). The derivatives of this data structure include layer, stack, matrix, table, vector, window, and scalar objects. In addition to providing functions that create, copy, modify, and destroy these data types, the routines also handle operations such as data type conversion, standard point functions (e.g., sin, cos, etc.), GIS point functions (e.g., diversity or density), single argument operations (e.g., not or sign), multiple argument operations (e.g., multiply or divide), and neighborhood functions (e.g., majority or convolve).

Block File Creation

The (epho) API supports the easy creation of LPS block structures. It is most likely to be used by applications that import other data formats into the block file format. The block structure stores all the information that is associated with a block of images. Some this data is created and maintained by the LPS libraries and some of the data is created and maintained by the BlockModelInterfaces DLLs.

Feature Space

The Feature Space (efsp) package contains all the feature space functions. These include the reading and writing of feature space nodes as children in .img files and as lists of feature space nodes for communication with the transformation process. The functions in this package will also convert the image data into feature space plots and map auxiliary files (thematic) into the feature space.

Hierarchical File Access

The Hierarchical File Access (ehfa) package provides an implementation of the ERDAS IMAGINE EHFA file format. This format maintains an object-oriented representation of data in a disk file through the use of a tree structure. All non-ASCII files used by ERDAS IMAGINE are in this format (e.g., .img, .ovr, etc.). The functions in this routine allow the programmer to identify, create, copy, open and close EHFA files; and locate, read, and write objects within an EHFA file. The standard ERDAS IMAGINE data objects (layers, map information, descriptor tables, etc.) can be accessed through higher level packages (eimg, edsc, etc.), so these functions are normally used to manipulate application-specific data objects within EHFA files.

Histogram and Lookup Table
The Histogram and Lookup Table (ehst) package provides various functions on histograms and lookup tables, primarily in support of the histogram tools and the contrast adjustment in the Viewer. Histogram, LUT, and breakpoint structures can be created, converted, and edited using these routines.

Image File Access [RasterFormats DLL]
The Image File Access (eimg) package provides tools to access and manipulate data in image files supported by the RasterFormats DLL Class, including ERDAS IMAGINE .img files. The package includes functions to read and write objects commonly found in geospatial raster datasets such as map information, projection parameters, statistics, histograms, covariance matrices, descriptor tables, class names, and color values. The package also provides high-level functions for processing the raster data within the dataset including functions to create, read, and write raster layers; associate windows, and areas of interest with a raster layer; and apply neighborhood functions to raster layers. It is through this package and its constructs that an application may modify the actual data values in a raster layer.

Image Resampling [ResampleMethods DLL]
The Image Resampling (eirf) package provides tools to resample imagery that is being geometrically transformed (see Transform). Three resampling methods are supported: nearest neighbor, bilinear interpolation, and cubic convolution. It is possible to support other types of resampling through the ResampleMethods DLL Class mechanism.

Kernel Library
The Kernel Library (eklb) package provides for reading and writing convolution filter kernels from a kernel library file.

Map Printing
The Map Printing (emap) package provides routines for generating and reading map files (.map) and for creating, reading, and writing all types of files associated with hardcopy map output.

Map Projection
The Map Projection (eprj) package provides tools for storing and retrieving map projection information for a layer and for converting coordinates between map projections. It also contains functions necessary for developing projections that are external to the ERDAS IMAGINE package.

Pixel Management
The Pixel Management (epxm) package provides functions for creating and copying rectangular arrays of pixels of any of the thirteen data types supported by ERDAS IMAGINE. The pixel rectangles are compatible with the egda functions.

Raster GIS Analysis
The Raster GIS Analysis (erga) package provides tools to analyze raster data. It includes tools for point analysis, neighborhood analysis, regional analysis, and feature extraction. This package is a high-level package that operates on top of the eimg package (unlike the egda, efga, and eirf packages which are not strictly dependent on any eimg constructs).

Seed/Region Growing
The Seed/Region Growing (esed) package performs the region growing functions. Functions exist at different levels to do the raster region grow, convert the raster region to a polygon, and generate an annotation layer of the region polygon.

Signature Management

The Signature (esig and esig_io) packages contain all the signature support functions. Esig provides functions to create signatures, set and get signature attributes, copy signatures, manage lists of signatures, merge signatures, etc. Esig_io contains all the file I/O functions for signatures and signature lists.

Statistics

The Statistics package (esta) provides tools for manipulating (i.e., reading and writing to/from files, creating and copying in-memory data structures, computing, etc.) statistics (mean, standard deviation, etc.), histograms, covariance matrices, color tables, and class names. This is a middle level package in that it is utilized by the eimg package but relies on the edsc, ehfa, and emif packages.

Vector

The Vector (evec) package provides tools for:

- Creating and deleting structures related to vector layers
- Opening and closing vector layers
- Reading features and their attributes
- Getting descriptive information about the layer
- Measuring distances and lengths
- Modifying vertices of arcs (splining, generalizing, and densifying)
- Determining the spatial relationship between points, lines, and polygons

Transform [GeometricModels DLL]

The Transform (exfr) package provides a generalized set of tools for creating and managing coordinate transforms, regardless of the type. Each transform may be a combination of any number of the basic transforms. The package provides functions for mapping arrays of points through a transform, inverting transforms, and composing transforms. The package is also constructed so that the programmer can create new types of transforms. This package is extensible through the use of the Geometric Model DLLs.

Application Environment Routines

Argument Parsing

The Argument Parsing (earg) package provides a convenient means of creating applications that are controlled by command line options. Parsing the command line and looking for options is a tedious task and without a set of standard tools, the resulting command line syntax will vary from program to program.

Error Logging and Reporting

The Error Logging and Reporting (eerr) package provides functions to create, delete, and print error reports. Nearly every function in the libraries of the IMAGINE Developers' Toolkit requires an argument that represents a place for the function to pass back an error report to the calling function. The functions and macros defined in this package will allow the programmer to work within the error reporting system of the IMAGINE Developers' Toolkit.

Master Initialization

The Master Initialization (eint) package allows the programmer to initialize the IMAGINE Developers' Toolkit. This step is required in an application before the other IMAGINE Developers' Toolkit functions can be used. Progress Meter Support The Progress Meter Support (emet) package allows an ERDAS IMAGINE application to report its progress in an extensible way. Placing calls to the emet package at critical points in a process allows a user to track the progress of an application.

Progress Meter Support

The Progress Meter Support (emet) package allows an ERDAS IMAGINE application to report its progress in an extensible way. Placing calls to the emet package at critical points in a process allows a user to track the progress of an application.

Magnetic Tape Access

The Magnetic Tape Access (emta) package provides a client-side API to the ERDAS IMAGINE tape server. Using the API to the tape server, an application can open, close, and manipulate (e.g., rewind, skip files) tape devices, as well as read and write data to those devices. The type of device (8 mm, track, etc.) and its location on the network (local, remote) are transparent to the API, and therefore, the application.

Preference Database

The Preference Database (epmg) package provides a programmer with an interface to the Preference Database. It provides functions to access, set, and change the preferences for different categories at global and user levels. It also provides callback mechanisms to notify the clients when a preference value is changed.

Session Manager

The esmg (Session Manager) package provides an interface to the ERDAS IMAGINE Session Manager. The primary functions are: connect to the Session Manager, indicate job state, indicate job progress, log warning/status messages in the Session Log, and indicate the end of a job.

Viewer

The Viewer (evue) package contains a set of functions for interaction with the ERDAS IMAGINE Viewer. These functions allow an application to do the following:

- Query the contents of specified viewers
- Load image, annotation, vector, or AOI layers into specified viewers
- Add callbacks on images in the viewer
- Create graphics representing lines or areas at specified locations
- Interactively select areas or lines
- Display compositions

- Set image extents
- Create/destroy viewers
- Warp images
- Create screen links
- Set convolution kernels

Low-Level File I/O and System Access Routines

Low Level File I/O

The Low Level File I/O (efio) package provides functions for creating, reading, writing, locating, and finding information about files and directories. Many of these functions are simply wrappers for existing POSIX and ANSI C functions. The wrapper allows for expansion of environment variables in file names and directory paths, as well as the translation of the standard 'errno' variable into an error report consistent with the eerr package.

Other functions allow temporary file names to be generated and for a directory path to be searched while attempting to locate a particular file. The package also allows efficient I/O that is portable across platforms.

Machine Independent Format

The Machine Independent Format (emif) package allows data to be read and written to disk in a format that is sharable across compilers, operating systems, and machine architectures. The package allows a programmer to create and destroy dictionaries of object designs, create and destroy object designs in those data dictionaries, and to read and write data to disk based on an object design. The standard ERDAS IMAGINE data objects (layers, map information, descriptor tables, etc.) can be accessed through higher-level packages (eimg, edsc, etc.), so these functions are normally used to manipulate application-specific data objects within EHFA files.

Timer

The Timer (etim) package provides routines for timing code execution.

Abstract Object Manipulation Routines

Binary Search Tree

The Binary Search Tree (ebst) package provides a complete set of tools for creating and maintaining sorted binary search trees. Modeled after the qsort/bsearch functions in the standard C libraries, the ebst package works with any type of programmer-supplied data for which the programmer provides a compare function.

Dynamic List Manager

The Dynamic List Manager (edlm) package provides a means to create and retrieve data from lists and stacks of arbitrary size. The lists, or stacks, may contain any type of object. Lists and stacks are useful in situations in which you need to build a set of data by sequentially adding objects, but you do not know beforehand how many objects will be in the set.

Linked List
The Linked List (ellm) package provides a common set of functions used to create and manage linked lists. The lists may contain any type of data.

Miscellaneous
The Miscellaneous (emsc) package defines macros and functions that deal with memory management or are otherwise not specific to any one routine in the IMAGINE Developers' Toolkit (e.g., EMSC_MIN, EMSC_MAX, EMSC_CURRENT_VER).

Selection
The Selection (esel) package provides a set of functions for implementing queries based on a simple query language. A query is then converted to an internal form, which is then used by the query evaluator. The result of the query is stored in a bitmapped table of Boolean results. There is a complete set of functions for managing the resulting Boolean tables.

String Manipulation
The String Manipulation (estr) package currently provides functions for six categories of character string manipulation:

- General string manipulation (compare, change, duplicate, analyze character strings)
- File name string parsing
- String list manipulation
- String expression evaluation
- Numeric formatting to a character string
- Miscellaneous string functions (e.g., converting errno to a string description of the error)

Symbol Table
The Symbol Table (esym) package provides a set of functions used to create and manage symbol tables. Each holds one or more symbols which relate a string with some type of user data. The package provides functions for adding symbols to the table and for locating symbols within a given table.

EML GUI Access Routines

EML
The EML (eeml) package provides a programmer with an interface to all the ERDAS Macro Language (EML) API functions. Application programs that present a GUI through an EML script should use the functions in this package. The APIs in this package will provide functions to do the following:

- Create a GUI for the application program using the EML scripts as templates
- Display/undisplay the dialog frames defined in the EML script
- Add/remove callback functions on the GUI parts defined in the EML script
- Manipulate GUI objects (get/set values, etc.)

Button
The Button (eeml_button) package provides functions specific to the EML button part. The APIs in this package will provide functions to:

- Get/set colors for color buttons
- Arm/disarm/toggle buttons

Canvas
The Canvas (eeml_canvas) package provides a systemindependent interface to on-screen and off-screen graphics primitives. Many of the primitives are specifically designed to work with the structures from the eimg package, making it easier to read ERDAS IMAGINE files and display the data to the screen. These primitives can be used to build graphic applications such as the ERDAS IMAGINE Viewer. Using this package in conjunction with the rest of the EML package will result in an application that is easily ported between the various operating systems and window systems that ERDAS IMAGINE supports.

Dialog
The Dialog (eeml_dialog) package contains functions that provide consistent interaction with the user for common user-interface operations such as asking the user for confirmation, displaying a warning message, displaying a status message, etc.

Frame
The Frame (eeml_frame) package provides functions specific to EML dialog frames. The APIs in this routine provide functions to:

- Set/get geometry to/from displayed or undisplayed frames
- Display the frame on a different screen (dual-headed systems)
- Set the title string, display the status message, show/hide the menu bar, show/hide the status bar, etc., access frame menus, etc.

Generic Part
The Generic Part (eeml_generic) package provides functions specific to the EML generic part. The generic part is designed as a space holder in an EML dialog. It typically reserves space in the dialog for a user interface part that cannot be specified in EML, such as a CellArray™ or a canvas. Aspects of the user interface that cannot be handled in EML must be handled programmatically. Through the eeml_generic portion of the eeml package, functions are made available to the programmer that provide a consistent and portable way to display a busy cursor over a generic part while that part is being updated.

Menu
The Menu (eeml_Menu) package provides the functions to create and interact with popup and pull down menu systems.

Popup List
The Popup List (eeml_popuplist) package provides functions specific to the EML popup list part. The APIs in this package will provide functions to:

- Replace the old list with a new list
- Replace the old picture list with new picture list

Projection Editor
The Projection Editor (eeml_prjeditor) package provides a function to control an interactive Projection Editor that can be used by application programs to get projection parameters from the user.

Scrolled List
The Scrolled List (eeml_scrollist) package contains functions specific to the EML scroll list part. The functions allow application programs to manipulate the EML scroll list (e.g., add items, replace items, etc.).

Tablet Digitizer
The Tablet Digitizer (eeml_tablet) package provides functions to access the Tablet Digitizer driver via an interactive GUI. The routine provides API functions to do the following:

- Open/connect to the tablet digitizer
- Add callbacks on the digitizer
- Close/disconnect from the tablet digitizer

Text
The Text (eeml_text) package contains functions specific to the EML text part. The functions will allow application programs to:

- Insert text
- Overwrite/replace/append text
- Turn on/off the highlight attribute of the text
- Get/set the cursor position on the text
- Cut/copy/paste text
- Set/get/clear the primary selection on the text

CellArray™
The CellArray (eeui_CA) package provides a single user interface for dealing with the management of tabular data. The programmer provides the description of the columns (string, double, color, etc.) and provides the functions to get and put the data. The CellArray package manages the navigation and display of any amount of data. Built-in functions allow search, sorting, and report generation.

Charts
The Chart (eeml_Chart) package provides a graphical part, which can be used to plot data in several modes. The chart is a tool that can plot tabular data as line graphs on a grid with an associated legend. This is the basis of the various profile tools in ERDAS IMAGINE.

ColorWheel
The Colorwheel (eeui_CW) package provides an HIS color part that can be used for color selection. A slider controls the intensity, with hue and saturation mapped to the angle around a circle and the distance from its center. To select colors, the user can drag a single point within the color wheel. The application is notified of color changes through a callback mechanism.

Histogram
The Histogram (eeui_Histogram) package provides an interactive tool for displaying and manipulating histogram data. The user may zoom in to get more detail on any area of the histogram. In addition, the

routine provides functions for controlling the contents of the plot as well as printing the histogram to a PostScript file.

New Capabilities

While the areas above have been extended with recent releases of the IMAGINE Developers' Toolkit, the routines below are completely new.

2D Visualization Routines

ViewWindow

The Viewer package provides a basic set of tools for embedding one or more viewers into an IMAGINE application. A viewer can display various types of 2D data including vectors, annotation, and images. These routines also provide a means of extending the types of data that can be displayed within the viewer via the ViewLayer class.

The ViewWindow (Edis_ViewWindow) package is responsible for managing the display area of the viewer. When a viewer is created, a handle to the Edis_ViewWindow is returned and can be used with subsequent calls to the Edis_ViewWindow API. This API can be used for toggling scroll bars, adding ViewLayers, scrolling imagery, etc.

ViewLayer

The ViewLayer (Edis_Viewlayer) package that can be used for manipulating ViewLayer objects within a ViewWindow. This API is used for closing ViewLayers, changing band combinations, setting zoom levels, etc. The API can be used with existing ViewLayer types such as the TrueColor, PseudoColor, GrayScale, Annotation, Vector, and AOI layers.

ViewLayer Class

The ViewLayer (Edis_ViewLayer) class defines the mechanism used for creating new types of ViewLayers that can be displayed within a ViewWindow. This becomes necessary when the ViewLayer types that are provided with the toolkit do not provide features required by the application.

Graphics Context/Target

The Graphics Context and Graphics Target API provide ViewLayers the mechanism needed to render them into the ViewWindow. When implementing new ViewLayer types using the Edis_ViewLayer class it will be necessary to utilize these functions. The Graphics Context is provided for setting attributes such as color, size, etc., and the Graphics Target is provided for doing the actual drawing.

Selector

The Selectors (Edis_SelNew) package provides an interface for creating graphical representations of objects that are used by the user for editing elements. The selectors do not provide a very robust set of display styles because they are typically used for displaying fast graphical elements including polylines, polygons, points, icons, and rectangles.

Selector Class

The selectors class provides a mechanism for new selector types to be created when the existing selectors do not provide the features required by the application.

Picks

The Picks API provides tools used when the user is responsible for creating an element interactively using the mouse. The picks have been used for creating annotation elements, AOI elements, selecting viewers, etc. The API returns the coordinates selected by the user to the caller.

Raster Edit

This API provides a few routines for modifying properties of raster images including contrast and convolutions kernels. These functions can also be used for modifying the pixel values in a TrueColoror GrayScale image layer.

About Hexagon Geospatial

Hexagon Geospatial helps you make sense of the dynamically changing world. Known globally as a maker of leading-edge technology, we enable our customers to easily transform their data into actionable information, shortening the lifecycle from the moment of change to action. Hexagon Geospatial provides the software products and platforms to a large variety of customers through direct sales, channel partners, and Hexagon businesses, including the underlying geospatial technology to drive Intergraph® Security, Government & Infrastructure (SG&I) industry solutions. Hexagon Geospatial is a division of Intergraph® Corporation. For more information, visit www.hexagongeospatial.com.

Intergraph® Corporation is part of Hexagon (Nordic exchange: HEXA B). Hexagon is a leading global provider of design, measurement and visualisation technologies that enable customers to design, measure and position objects, and process and present data.

Learn more at www.hexagon.com.