Loading

Creation

Storage

1. Canvas Operations

A canvas is the primary storage type in EZ-ASCII. All of the image modification happens on this type. Internally, it is represented as a two-dimensional array of integers referred to as intensities. This canvas can be loaded from an existing image file or it can be created manually.

* 1. Intensity Mapping and Granularity

The intensity mapping contains a table of intensities to characters. The is done to simplify a programmers interaction with the canvas. A programmer should not have to consider

which exact character they are drawing but instead just how intense each point should be. This makes drawing much simpler. Each intensity mapping also has an associated *granularity* as defined in section X.Y. **DEFAULT**.

A custom mapping can be defined in the following way:

MAP <- {I0:C0, I1:C1, … ,IN:CN}

*MAP* is a keyword referring to the intensity map. Each *I* is an intensity and the corresponding *C* is the character mapped to that intensity. Any reference to the intensity mapping will reference to the most recent assignment of *MAP* or the default if none have been assigned.

* 1. Creation

The are two ways to create a Canvas in EZ-ASCII. One is to take an image and load it.

* 1. Printing

Using the -> operator, a canvas can be saved to a file or printed to a console output. The effect of this operator is to apply the intensity mapping to the canvas and print the corresponding canvas to console or a file.

Saving to a file can be done using the following syntax:

canvas -> file\_path, render

The *canvas* must be a valid identifier of a canvas object as described by section *Y.Z. render* must be a boolean. If render is true, the intensity mapping will be applied to the canvas before printing and the end result will be a file of characters. If render is false, the canvas will be printed as is The file name must be a path to a valid file location in the form of string type. If the file does not exist, it will be created. and the contents of the canvas, mapped into the

appropriate characters will be printed to it. If the file already exists, it will be overridden.

Printing to the console can be done in a similar way:

canvas -> out, render

Again, the *canvas* must be a valid identifier of a canvas object and render must be a valid boolean value. o*ut* refers to the out keyword described in section Y.Z. In this case, the contents of the map will be printed to standard output instead of a File.