Grace Package Manager Specification

I BASIC CONCEPTS

Package in grace:

> A bundle of .grace files together with a package descriptor.

II OVERVIEW

Grace Package Manager to be a standalone tool that can be run from the command line (and may have a graphical interface in the future). Gpm can be given a range of command line arguments and options.

gpm list	- lists currently installed packages
gpm get	- downloads a package or file and all dependencies
gpm build	- installs package from specified url
gpm install	- updates package from specified url
gpm bundle	- generates package and package descriptor
gpm help	- lists command line arguments

Broadly modelled on Go package manager and node.js npm.

III SPECIFIC FUNCTIONALITY

A. get

get [file]

> *gpm get dogepath.com/wowfiles/wow.grace*

In this example the package manager will download wow.grace, work through all files imported by wow.grace, and recursively download all necessary dependencies (unless already installed on the local machine).

Each downloaded file will be placed in a directory structure in .local/lib/share/[package] that mirrors the download location.

get [package]

> gpm get github.com/coolpackage/cool.tar.gz

1. Downloads and unpacks tarball

2. Package descriptor file required to be in package tarball. Package descriptor (pkg.grace) is a dialect that returns a module containing the necessary information for the package:
    - name
    - Exectuable Info
        - main file (compiler.grace)
        - compilation options (--module)
        - external dependencies
    - Installation Info
        - src
        - external dependencies.

get -build [package]

gpm get -build github.com/coolpackage/cool.tar.gz

1. Downloads remote package (if not remote, get not required here)

2. If new package, unpack package into usr/lib/grace/[PACKAGE NAME] where the package name is defined by the package descriptor.

3. Evaluates package descriptor to determine additional dependencies.

4. Compiles modules within tarball, along with external dependencies, into dist/exec/ $name and dist/lib/$name folders.

Note: does not install dependencies into global libraries.

get -install package]

gpm get -install github.com/coolpackage/cool.tar.gz

1. Downloads remote package

2. Copies files from dist/exec and dist/lib into $HOME/.grace/bin/minigrace/$NAME $HOME/.grace/lib/grace-modules/$NAME respectively.

Note: install and build can be applied as separate commands that can be applied to a previously downloaded (non remote) tarball

B. bundle

bundle [directory]

Example:

%gpm bundle [package]

1. Takes folder with pkg.grace file and new package name as arguments. If pkg.grace file not found, error displayed.

2. Default option pulls in all local modules (that is, modules required by .grace files within folder than are located on local system).

3. "–static" option pulls in all local AND remote dependencies (e.g. http://dep.com/dep.grace). Checks pkg.grace to ensure found dependencies match those specified.

4. Tars folder and all additional modules/dependencies and pkg.grace file into single package that can be downloaded and installed.

C. list

Example:

%gpm list

1. Lists all installed packages

Minigrace: The Library

- Minigrace currently opens multiple instances of the compiler program to perform execution of multiple tasks.
- Compiler stores global state in util.grace.
- In order to enable minigrace to be imported as a library, this global state needs to be decouple.