

Example

Example/Template of a GAP Package

4.4.1

20 June 2025

Werner Nickel

Greg Gamble

Olexandr Konovalov

Werner Nickel

Homepage: <http://www.mathematik.tu-darmstadt.de/~nickel>

Greg Gamble

Email: gregg@math.rwth-aachen.de

Homepage: <http://www.math.rwth-aachen.de/~Greg.Gamble>

Address: Greg Gamble

Department of Mathematics and Statistics
Curtin University of Technology
GPO Box U 1987
Perth WA 6845
Australia

Olexandr Konovalov

Email: obk1@st-andrews.ac.uk

Homepage: <https://olexandr-konovalov.github.io/>

Address: School of Computer Science

University of St Andrews
Jack Cole Building, North Haugh,
St Andrews, Fife, KY16 9SX, Scotland

Copyright

© 1997–2022 by Werner Nickel, Greg Gamble and Olexandr Kononov

`Example` package 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.

Acknowledgements

We appreciate very much all past and future comments, suggestions and contributions to this package and its documentation provided by `GAP` users and developers.

Contents

1	The Example Package	4
1.1	The Main Functions	4
2	Installing and Loading the Example Package	6
2.1	Unpacking the Example Package	6
2.2	Compiling Binaries of the Example Package	6
2.3	Loading the Example Package	7
	Index	8

Chapter 1

The Example Package

This chapter describes the GAP package `Example`. As its name suggests it is an example of how to create a GAP package. It has little functionality except for being a package.

See Sections 2.1, 2.2 and 2.3 for how to install, compile and load the `Example` package.

If you are interested in developing a GAP package, see (**Reference: Using and Developing GAP Packages**).

If you are viewing this with on-line help, type:

```
gap> ?Example package
```

to see the functions provided by the `Example` package.

1.1 The Main Functions

The following functions are available:

1.1.1 ListDirectory

▷ `ListDirectory([dir])` (function)

lists the files in directory *dir* (a string) or the current directory if called with no arguments.

1.1.2 FindFile

▷ `FindFile(directory_name, file_name)` (function)

searches for the file *file_name* in the directory tree rooted at *directory_name* and returns the absolute path names of all occurrences of this file as a list of strings.

1.1.3 LoadedPackages

▷ `LoadedPackages()` (function)

returns a list with the names of the packages that have been loaded so far. All this does is execute

Example

```
gap> RecNames( GAPInfo.PackagesLoaded );
```

You might like to check out some of the other information in the GAPInfo record (see **(Reference: GAPInfo)**).

1.1.4 Which

▷ Which(*prg*) (function)

returns the path of the program executed if Exec(*prg*); is called, e.g.

Example

```
gap> Which("date");
"/bin/date"
gap> Exec("date");
Fri 28 Jan 2011 16:22:53 GMT
```

1.1.5 WhereIsPkgProgram

▷ WhereIsPkgProgram(*prg*) (function)

returns a list of paths of programs with name *prg* in the current packages loaded. Try:

Example

```
gap> WhereIsPkgProgram( "hello" );
```

1.1.6 HelloWorld

▷ HelloWorld() (function)

executes the C program hello provided by the Example package.

1.1.7 FruitCake

▷ FruitCake (global variable)

is a record with the bits and pieces needed to make a boiled fruit cake. Its fields satisfy the criteria for Recipe ([1.1.8](#)).

1.1.8 Recipe

▷ Recipe(*cake*) (operation)

displays the recipe for cooking *cake*, where *cake* is a record satisfying certain criteria explained here: its recognised fields are name (a string giving the type of cake or cooked item), ovenTemp (a string), cookingTime (a string), ingredients (a list of strings each containing an _ which is used to line up the entries and is replaced by a blank), method (a list of steps, each of which is a string or list of strings), and notes (a list of strings). The global variable FruitCake ([1.1.7](#)) provides an example of such a string.

Chapter 2

Installing and Loading the Example Package

2.1 Unpacking the Example Package

If the Example package was obtained as a part of the GAP distribution from the “Download” section of the GAP website, you may proceed to Section 2.2. Alternatively, the Example package may be installed using a separate archive, for example, for an update or an installation in a non-default location (see (**Reference: GAP Root Directories**)).

Below we describe the installation procedure for the `.tar.gz` archive format. Installation using other archive formats is performed in a similar way.

To install the Example package, unpack the archive file, which should have a name of form `example-XXX.tar.gz` for some version number `XXX`, by typing

```
gzip -dc example-XXX.tar.gz | tar xpv
```

It may be unpacked in one of the following locations:

- in the `pkg` directory of your GAP 4 installation;
- or in a directory named `.gap/pkg` in your home directory (to be added to the GAP root directory unless GAP is started with `-r` option);
- or in a directory named `pkg` in another directory of your choice (e.g. in the directory `mygap` in your home directory).

In the latter case one must start GAP with the `-l` option, e.g. if your private `pkg` directory is a subdirectory of `mygap` in your home directory you might type:

```
gap -l ";myhomedir/mygap"
```

where `myhomedir` is the path to your home directory, which may be replaced by a tilde (the empty path before the semicolon is filled in by the default path of the GAP 4 home directory).

2.2 Compiling Binaries of the Example Package

After unpacking the archive, go to the newly created `example` directory and call `./configure` to use the default `../..` path to the GAP home directory or `./configure path` where `path` is the path to the GAP home directory, if the package is being installed in a non-default location. So for example if

you install the package in the `~/.gap/pkg` directory and the **GAP** home directory is `~/gap4r5` then you have to call

Example

```
./configure ../../../../gap4r5/
```

This will fetch the architecture type for which **GAP** has been compiled last and create a `Makefile`. Now simply call

Example

```
make
```

to compile the binary and to install it in the appropriate place.

2.3 Loading the Example Package

To use the **Example Package** you have to request it explicitly. This is done by calling `LoadPackage` (**Reference: LoadPackage**):

Example

```
gap> LoadPackage("example");
-----
Loading Example 4.4.0 (Example/Template of a GAP Package)
by Werner Nickel (http://www.mathematik.tu-darmstadt.de/~nickel),
  Greg Gamble (http://www.math.rwth-aachen.de/~Greg.Gamble), and
  Olexandr Konovalov (https://olexandr-konovalov.github.io/).
maintained by:
  Greg Gamble (http://www.math.rwth-aachen.de/~Greg.Gamble) and
  Olexandr Konovalov (https://olexandr-konovalov.github.io/).
Homepage: https://gap-packages.github.io/example
Report issues at https://github.com/gap-packages/example/issues
-----
true
```

If **GAP** cannot find a working binary, the call to `LoadPackage` will still succeed but a warning is issued informing that the `HelloWorld()` function will be unavailable.

If you want to load the **Example** package by default, you can put the `LoadPackage` command into your `gaprc` file (see Section **(Reference: The gap.ini and gaprc files)**).

Index

Example package, [4](#)

FindFile, [4](#)

FruitCake, [5](#)

HelloWorld, [5](#)

License, [2](#)

ListDirectory, [4](#)

LoadedPackages, [4](#)

Recipe, [5](#)

WhereIsPkgProgram, [5](#)

Which, [5](#)