Strand Development Kit

Getting Started: Hello World

Run the `hello.str` example which just prints "Hello world!"

You can run this right here using the command line:

```
bin\strands examples\hello.str
```

Whose output is just

```
Hello world!
```

You can also run it from within the `examples` directory, like this:

```
..\bin\strands hello.str
```

Let's have a look at `hello.str`, to see how it works:

```
STORY
Hello world!
```

`Strands` is an IF authoring language based on the idea of *flow.* The flow starts at the beginning and continues until there is nowhere to go, at which point it stops, and the game is over.

Anything in capital letters is a *term.* Here, `STORY` is a term. When flow encounters a term, it flows *into* the term and comes back once the term stops.

So here we have a term called `STORY`, whose flow is the text "Hello world!". When flow encounters `STORY`, the words "Hello world!" will always be emitted. Since `STORY` is the first term in the file, flow starts here.

There is nothing special about using the word `STORY`. For example, the following would work equally as well.

```
START
Hello world!
```

Now we know about terms and flow, we could use our term twice:

```
START
STORY and STORY

STORY
Hello world!
```

This is in `examples/hellohello.str`

Run it!

```
bin\strands examples/hellohello.str
```

Output:

```
Hello world! and Hello world!
```

Flow starts at `START` simply because it's the first term in the file. Flow immediately encounters `STORY`, which emits the text "Hello world!", after which flow returns and find the text "and" which it emits, then it encounters another `STORY`, which results in "Hello world!" being emitted once again.

By using flow, you create a game. For documentation on flow, terms and how to build games see the manual in `doc\strandmanual.pdf`

Running the Examples

These examples are also covered in the `strandmanual.pdf`.

Running Beanstalk

Go into the directory `examples\beanstalk`

Run the command-line version with:

`go.bat`

Run the web version with

`goweb.bat`

The latter will copy the runtime and story into a local `web` directory, then launch your browser.

Running Picton

Go into the directory `examples\picton`

Run the command-line version with:

`go.bat`

Run the web version with

`goweb.bat`

The latter will copy the runtime and story into a local `web` directory, then launch your browser.

How to make a new game

The `core` directory contains the standard `Strands` library. This is a set of term definitions that have been put into `core.str` to get games going quickly.

For the following, we'll assume your game name is "mygame"

Step 1

Make your game directory and copy in the `core` files:

```
cd examples
mkdir mygame
cd mygame
copy ..\..\core\*
```

Now you can run the game already!

`go.bat`

Will give you a small two-room game, that's defined in `map.str`. So now to customise.

Step 2

Rename some files and setup `story.str`:

```
move game.str mygame.str
```

edit `story.str` as follows and change `GAME_FILES`:

```
GAME_FILES
core.str map.str mygame.str
```

While editing `story.str` also fill in your versions of;

```
GAME_TITLE
The Game

GAME_AUTHOR
by A.Hacker

GAME_VERSION
1.0
```

Step 3

Make `map.str`

The existing `map.str` is just a two location dummy. Need to expand this to be the locations of your game.

Also make sure you set the player in the first location. This was done in the template `game.str` file here:

```
BEGIN
\nThe game begins.
UPDATEMAP
GOHALL
MAIN
```

`GOHALL` referenced the term to put you in the start location. Change this as appropriate for your `map.str`.

Step 4

Run it!

`go.bat` will run the console version

`goweb.bat` will copy in the web runtime and launch your game in a browser.