Vim Interaction

The plugin presents a function called `callvim` whose usage is:

```
usage: callvim [-b cmd] [-a cmd] [file ... fileN]

  -b cmd     Run this command in GVIM before editing the first file
  -a cmd     Run this command in GVIM after editing the first file
  file       The file to edit
  ... fileN  The other files to add to the argslist
```

Rationale

The idea for this script is to give you some decent interaction with a running GVim session. Normally you'll be running around your filesystem doing any number of amazing things and you'll need to load some files into GVim for editing, inspecting, destruction, or other bits of mayhem. This script lets you do that.

Aliases

There are a few aliases presented as well:

- `v` A shorthand for `callvim`
- `vvsp` Edits the passed in file but first makes a vertical split
- `vhsp` Edits the passed in file but first makes a horizontal split

Post Callout

At the end of the `callvim` function we invoke the `postCallVim` function if it exists. If you're using MacVim, for example, you could define a function that brings window focus to it after the file is loaded:

```
function postCallVim
{
  osascript -e 'tell application "MacVim" to activate'
}
```

This'll be different depending on your OS / Window Manager.

Examples

This will load `/tmp/myfile.scala` into the running GVim session:

```
> v /tmp/myfile.scala
```

This will load it after first doing a vertical split:

```
> vvsp /tmp/myfile.scala
or
> v -b':vsp' /tmp/myfile.scala
```

This will load it after doing a horizontal split, then moving to the bottom of the file:

```
> vhsp -aG /tmp/myfile.scala
or
> v -b':sp' -aG /tmp/myfile.scala
```

This will load the file and then copy the first line to the end (Why you would ever want to do this... I dunno):

```
> v -a':1t$' /tmp/myfile.scala
```

And this will load all of the `*.txt` files into the args list:

```
> v *.txt
```

If you want to load files into areas that are already split, use one of the aliases for that:

```
# Do a ':wincmd h' first
> vh /tmp/myfile.scala

# Do a ':wincmd j' first
> vj /tmp/myfile.scala

# Do a ':wincmd k' first
> vk /tmp/myfile.scala

# Do a ':wincmd l' first
> vl /tmp/myfile.scala
```