



Public

Log In

- Public Web
- Create New Topic
- Index
- Search
- Changes
- Notifications
- Statistics
- Preferences

Webs

- Public
- TWiki

TWiki > Public Web > DpHipe > DpHipePerspectives (2014-02-19, AlvarGarcia)

Edit Attach

Tags:

+ [create new tag](#) [view all tags](#)

PDF Version [Portrait](#) [Landscape](#)

Did you spot something wrong or missing on this page? If you have an account on this TWiki, you can fix it yourself by editing the page. If you don't have an account, you can leave a message at the bottom of the page to tell us about it. Thank you in advance!

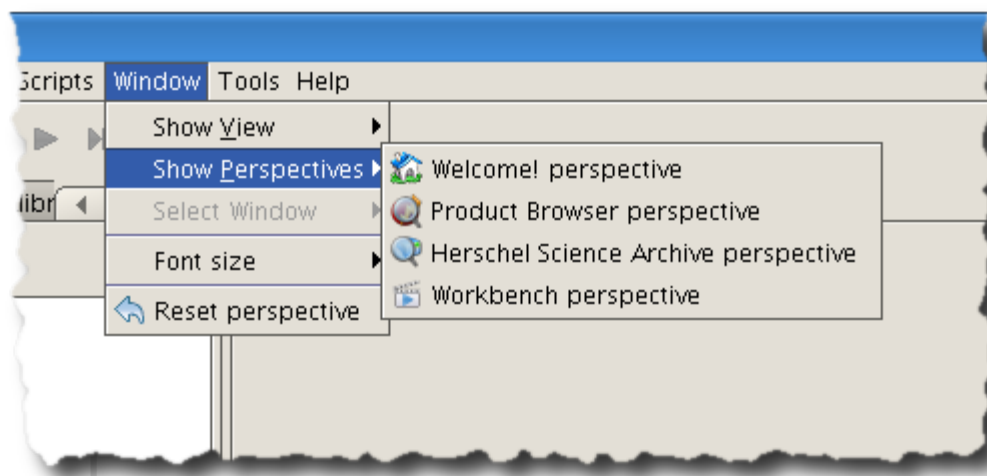
Creating a Perspective

If you think that a set of (new) [views](#) should be presented in standard layout, you can do so by creating a new perspective. Such a perspective defines how these views are laid out in tabs and split panes. This section briefly explains how to contribute a new perspective.

See also the [overview](#) page.

- ↓ [Basic Steps](#)
- ↓ [Construction items](#)
- ↓ [Example](#)
 - ↓ [Implementation](#)
 - ↓ [Registry](#)

Basic Steps



The aim is that your contribution is automatically picked up by the [infrastructure](#), such that your perspective is added to the

perspective menu, as shown in the picture on the right.

To make this happen, you will have to:

- Implement the [Perspective](#) interface.
- Register your implementation to the [Extension Registry](#) in the *init.py* file.

For your convenience an abstract implementation exists ([AbstractPerspective](#)); it allows you to concentrate on implementing one method only:

```
public abstract class AbstractPerspective implements
Perspective {

    public abstract SitePart doLayout(SitePartBuilder
builder);

}
```

Construction items

[SitePart](#) objects are the building blocks of a perspective. The following flavours are available:

- [ViewPart](#): normal views.
- [EditorArea](#): the editor area.
- [SplitPart](#): two windows with a movable split.
- [TabbedPart](#): several windows organized into tabs.

Access to the views, editor area or creation of the tab and split layout containers are all through the [SitePartBuilder](#) interface:

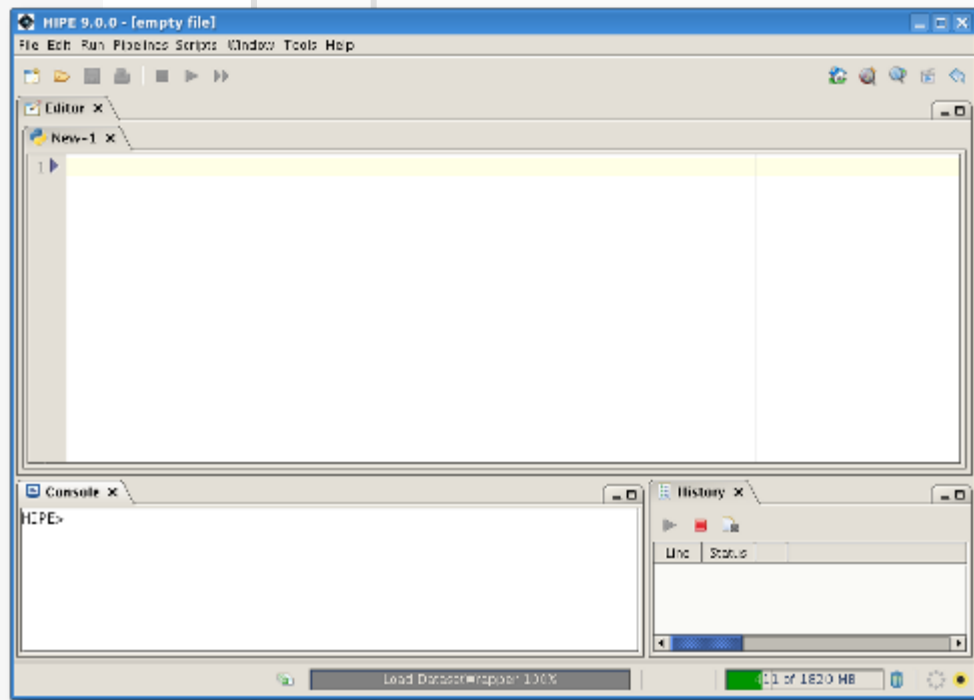
```
public interface SitePartBuilder {
    ViewPartManager getViewManager();
    TabbedPart buildTabs(SitePart... parts);
    SplitPart buildSplit(float split, boolean horizontal,
SitePart lhs, SitePart rhs);
}
```

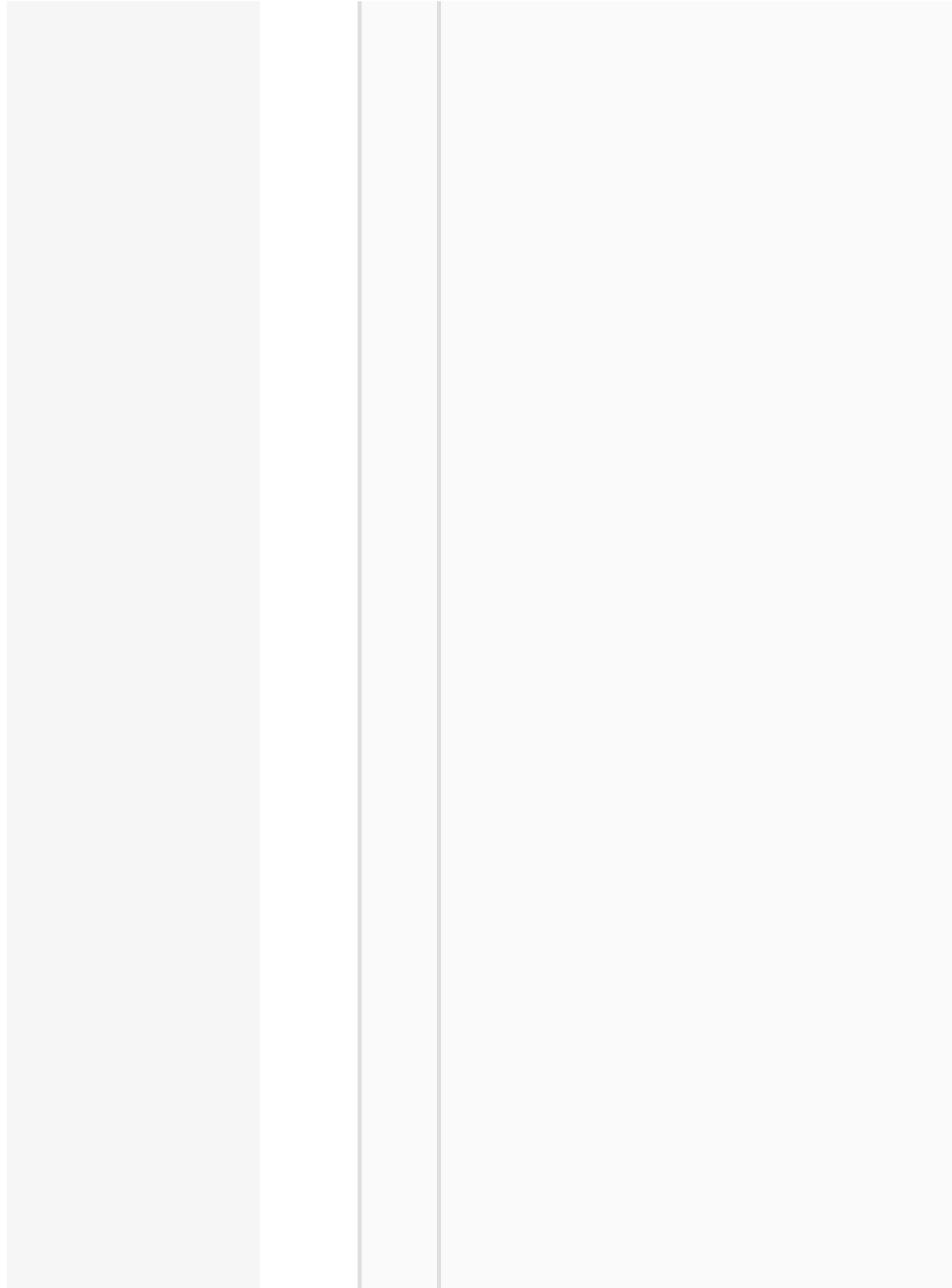
Example

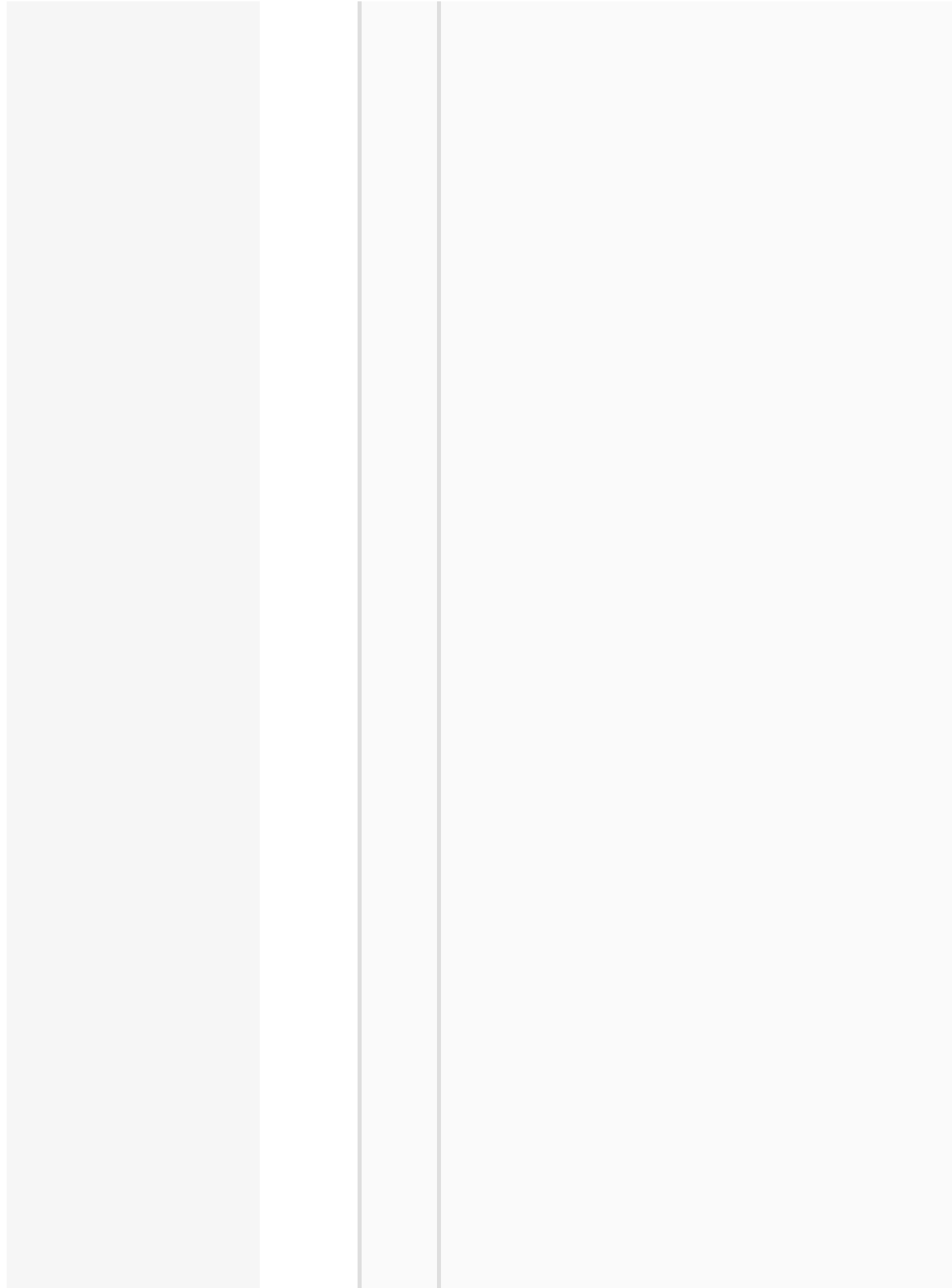
The above is explained by an example in which we are going to create a simple perspective with just three views. Next we are going to register it to [HIPE](#).

Implementation

The implementation below is made of three views: *Editor*, *Console* and *History*.







The above code is divided into three stages:

- Getting hold of the views you are interested in. This is done by asking the builder for the view manager. Then you grab existing views either by *title* or by a view's unique *identifier*.
- Creation of split panes that hold the views. First we create the bottom part (*console* and *history*), and then split the window into a top part (being the *editor area*) and the bottom part we just created.

The last part is returning the created site part.

Registry

The following snippet, could then go into your `__init__.py`:

```
REGISTRY.register(PERSPECTIVE, Extension(  
    "site.perspective.mysimple",  
    # a unique ID  
  
    "resides.in.java.package.MySimplePerspective"  
    # the above implementation  
    "My Simple",  
    # how it should be called in the  
menu  
    None  
    # possible icon (none here)  
))
```

See also the [Extension Registry](#) documentation for more details.

[blog comments powered by Disqus](#)

 [Edit](#) | [Attach](#) | [Watch](#) | [Print version](#) | [History](#):

[r21](#) < [r20](#) < [r19](#) < [r18](#) < [r17](#) | [Backlinks](#) | [Raw View](#) | [Raw edit](#) | [More topic actions](#)

Topic revision: [r21](#) - 2014-02-19 - [AlvarGarcia](#)

