Implementing the Tweet\Value class

The Value class of a Field Type is by design very simple. It is meant to be stateless, and as lightweight as possible. Therefore, this class must contain as little logic as possible, as this is the responsibility of the Type class .

The Value will at least contain:

- public properties, used to store the actual data
- an implementation of the `__toString()` method (required by the Value interface we inherit from)

By default, the constructor from `FieldType\Value` will be used, and allows you to pass a hash of property/value pairs. In our example, we can see that we can override it as well if we want.

The Tweet Field Type is going to store three things:

- The tweet's URL
- The tweet's author URL
- The body, as an HTML string

At this point, we don't care where those are stored. All we care about is what we want our Field Type to expose as an API. We end up with the following properties:

eZ/FieldType/Tweet/Value.php

```
/**
* Tweet URL on twitter.com (http://twitter.com/UserName/status/id).
* @var string
*/
public $url;


/**
* Author's tweet URL (http://twitter.com/UserName)
* @var string
*/
public $authorUrl;


/**
* The tweet's embed HTML
* @var string
*/
public $contents;
```

The only thing left to honor the `FieldType\Value` interface is the `__toString()` method. Let's say that ours will return the tweet's URL:

eZ/FieldType/Tweet/Value.php

```
public function __toString()
{
   return $this->url;
}
```