Development-Tips and Tricks

URL's

- wocontext.request().uri() = the URL currently being requested

There are several different URL's associated with your application, all of which can be retrieved from various methods on WOApplication. Here is a quick cheat sheet of them:

- WOApplication.application().applicationBaseURL() = /WebObjects
- WOApplication.application().baseURL() = /WebObjects
- WOApplication.application().cgiAdaptorURL() = http://hostname/cgi-bin/WebObjects
- WOApplication.application().directConnectURL() = http://hostname:port/cgi-bin/WebObjects/MyApplication.woa
- WOApplication.application().frameworksBaseURL() = /WebObjects/Frameworks
- WOApplication.application().host() = hostname
- WOApplication.application().servletConnectURL() = http://hostname/cgi-bin/WebObjects/MyApplication.woa
- WOApplication.application().webserverConnectURL() = http://hostname/cgi-bin/WebObjects/MyApplication.woa/-port

Browser IP

```
/** Returns the IP address of the client.
    * This should return accurate information whether in direct connect or
webserver deployment mode.
    * If performance caching is turned on on OS X server, this method will
correctly use pc-remote-addr
    * @return The IP address as a string.
    */
   public static String clientIP(WORequest request) {
       Object ipAddress = request.headerForKey("pc-remote-addr");
       if (ipAddress == null) {
               ipAddress = request.headerForKey("remote_addr");
               if( ipAddress == null ) {
                   ipAddress = request.headerForKey("remote_host");
                   if( ipAddress == null ) {
                       ipAddress = request._remoteAddress();
                       if( ipAddress == null ) {
                           ipAddress = request._originatingAddress();
                           if( ipAddress != null ) ipAddress =
((InetAddress)ipAddress).getHostAddress();
                       }
                   }
               }
       }
       return ipAddress == null ? "<address unknown>" :
ipAddress.toString();
   }
```

NSArray

It's in the docs, but NSArray's implementation of KeyValueCoding is not really what I was expecting. To get an object at a specific numeric index of an NSArray, you'd use the

```
objectAtIndex()
```

method. So what does

```
NSArray.valueForKey(String key)
```

return?

Well, first read the docs: file:///OSX/Developer/Documentation/WebObjects/Reference/com/webobjects/foundation/NSArray.html#valueForKey(java.lang.String)

It turns out that calling valueForKey on an array is the same as calling valueForKey for each element of that array. So if you have an NSArray of Users, calling valueForKey("email"); will return an NSArray of email addresses. calling valueForKey("documents"); will return an NSArray of NSArrays containing document objects. In hindsight (and from looking at the way WOBuilder handles key paths for arrays) this is kind of obvious. But I think the real lesson here is that it is easy to ignore the docs towards the end of an alphabetical page...

HTML-friendly String Truncating

```
import org.apache.commons.lang.*; //From Apache
 import org.clapper.util.text.*; // From http://www.clapper.org/

 public static String stripHTMLTagsAndConcatenate(String htmlString, int
numberOfChar) {
   return
(StringUtils.substringBeforeLast(StringUtils.abbreviate((HTMLUtil.stripHTM
LTags(htmlString)), numberOfChar), " ")) + "...";
 }
```