EROpenID Framework

Using EROpenID with Google Apps Federated login

Sample delegate for talking to Google Apps via Google's openID deferated login. Note Google does not support SRegRequest.

```
public class GoogleAppsDelegate extends EROpenIDManager.DefaultDelegate {

 @Override
 public List<MessageExtension> createFetchMessageExtensions(String
userSuppliedString, WORequest request,
   WOContext context) throws MessageException {
  ArrayList<MessageExtension> exts = new ArrayList<MessageExtension>();
  FetchRequest fetchRequest = FetchRequest.createFetchRequest();
  fetchRequest.addAttribute("Email", "http://axschema.org/contact/email",
true);
  exts.add(fetchRequest);
  return exts;
 }

}
```

Then override appendToResponse your standard login page with something like this:

```
public void appendToResponse(WOResponse r, WOContext c) {
  // Assume client and realm exists.
  if (client.doesOpenID()) {
   String url = ((ERXWOContext)
c).directActionURLForActionNamed(ERODirectAction.class.getName()
     + "/openIDRequest?" + "identity=" + client.openIdIdentity() +
"&realm="
     + realm, null);
   r.setHeader(url, "location");
   r.setStatus(302);
  } else {
   super.appendToResponse(r, c);
  }
 }
```

Notes: the realm needs to be something static and explicit to your app. We use this:

WOApplication.application().webserverConnectURL().replaceFirst("http", "https")

client.openIDIdentity() should return something like this:

clientsdomainname.com/openid

where openid is an XRDS document. comprehensive instructions on setting up the rest is here:

http://jeremiahlee.com/blog/2009/09/28/how-to-setup-openid-with-google-apps/