How to use Apache Droids Hello crawler?

Table of contents

1 Hello crawler

Note:

Please note that we do not crawl yet images, css and scripts linked in the pages, but this feature is on top of the TODO list (right next of the creation of a sample droid). We will try to implement it ASAP.

Remember the build.properties from the build process? This file controlls the behaviour of the Hello crawler. This properties can be overriden via custom calls of the different methods in your custom java code.

The HelloCrawler offers you a wget style robot. We will now describe the basic functionality. First it will open the initial webpage (with the `protocol plugin` corresponding to the uri) and tries to extract the outlinks (with the `parse plugin` for the corresponding content type). Then it will test the links found on the page with the regular expression defined in the regex file (via the `filter plugin`). If we find new links that are accepted by the filter we then merge them with the queue. The last step is to pass the input stream to the stack of . `handler plugins`

Like you can see the Hello droid crawler is build by various different type of plugin. At the moment we support the following types:

- protocol plugins
- parse plugins
- filter plugins
- handler plugins

1.1 Crawling

If you have set the property "droids.initial.url" to your initial url and edited the regex file "droids.filter.regex" to exclude/include follow up links, the Hello Droid will save the crawl to "droids.filter.regex" after you invoke:

```
ant droids.crawl
```

If you have not changed anything the result will be stored in $DROIDS_HOME/export like:

```
export/
`-- target-x.de
  |-- about.html
  |-- index.html
  |-- open.html
  `-- search.html
```

1.2 Ant target

To use this target in your ant based application you can add the following snippet to your build file and do not forget to implemented the **PROPERTIES** such as e.g. droids.name!

```
<target name="droids.crawl" description="--> Will invoke the crawling.">
  <java classname="org.apache.droids.Cli" fork="true" dir="${build.dir}"
    maxmemory="${droids.maxmemory}" failonerror="true"
    resultproperty="buildResult">
    <arg value="${droids.name}"/>
    <arg value="${droids.spring.context}"/>
    <classpath>
      <path refid="droids.classpath"/>
    </classpath>
  </java>
</target>
```

1.3 Java class

You can directly use the class `org.apache.droids.Cli` by calling the `main` method. There are two input parameters that you can use.

1. The droid name is obligatory. In the default configuration that is `Hello`
2. The second one is the location of the spring configuration you want to use. In the default configuration that is `classpath:/org/apache/droids/droids-core-context.xml`. This parameter is not obligatory. and if it is not provided we fall back to the above location.