Properties

Your Game Code - Adding Custom Elements · https://namazustudios.com/docs/custom-code/properties/

These are the various properties that can be added to your properties file or added to your custom Element code.

The properties file would be in the custom Element deployment directory, named `dev.getelements.element.attributes.properties`. The properties below can be defined or overridden within this file. For example:

`dev.getelements.elements.app.serve.prefix=example-element`

would cause the deployed Element to be given the name "example-element" and any endpoint defined within the Element would serve at /app/rest/example-element/<endpoint>

You can also define your own properties and inject them using the @Named annotation, like so

```
@Inject
public void setDefaultUserName(@Named(DEFAULT_USER_NAME) String defaultUserName) {
    this.defaultUserName = defaultUserName;
}
```

Note

To avoid potential conflicts, you should name properties using "com.mycompany" or whatever your domain is, instead of "dev.getelements"

For more information on how to structure a deployment, see Deploying an Element.

Default Root User Settings

```
/**
* The email/username to use for the default user.
*/
@ElementDefaultAttribute("root")
```

```
public static final String DEFAULT_USER_NAME = "dev.getelements.elements.user.default.name";

/**
 * The email/username to use for the default user.
 */
@ElementDefaultAttribute("root@example.com")
public static final String DEFAULT_USER_EMAIL = "dev.getelements.elements.user.default.email";

/**
 * The email/username to use for the default user.
 */
@ElementDefaultAttribute("example")
public static final String DEFAULT_USER_PASSWORD =
"dev.getelements.elements.user.default.password";
```

Element Settings

```
/**
* Defines an attribute which specifies the prefix for the element. At
* load-time, loader will inspect the attributes from the Element.
* If blank, then the loader will defer to the value of the
* ElementDefinition, which would typically be the name of the package
* bearing the annotation.
*
* This attribute forces the name of the Element to the assigned value when
* running locally in the IDE, which determines the path as well.
*
* For example, the below value will produce the relative path
* /app/rest/example-element, which any custom endpoints will append to.
*/
@ElementDefaultAttribute("")
String APPLICATION_PREFIX = "dev.getelements.elements.app.serve.prefix";
```

```
/**
* Defines an attribute which specifies if the elements should enable the
* standard auth pipeline in Elements.
* This ensures that the application server will be able to authenticate
* users using the Authorization or Elements-SessionSecret headers as well
```

```
* as allow the appropriate override headers to be used.
*
* If set to "true", the application will use the built in authentication
* service.
* If set to "false", the application will not use the authentication
* service and you will need to provide your own.
*
* Specifically when the dev.getelements.elements.auth.enabled attribute is
* set to "true", we will automatically install a set of filters that will
* ensure that the user is authenticated and applied to the service layer
* for all requests.
*/
@ElementDefaultAttribute("")
String ENABLE_ELEMENTS_AUTH = "dev.getelements.elements.auth.enabled";
```

Advanced Options

```
/**
* The storage directory for the git repositories housing the application's
* script storage.
*/
@ElementDefaultAttribute("cdn-repos/git")
public static final String GIT_CDN_STORAGE_DIRECTORY =
"dev.getelements.elements.git.cdn.storage.directory";

/**
* The storage directory for the git repositories housing the application's
* script storage.
*/
@ElementDefaultAttribute("script-repos/git")
public static final String ELEMENT_STORAGE_DIRECTORY =
"dev.getelements.elements.rt.git.element.storage.directory";

/**
* Used to specify the RPC provider for bsc blockchain.
*/
@ElementDefaultAttribute("https://data-seed-prebsc-1-s1.binance.org:8545")
String BSC_RPC_PROVIDER = "dev.getelements.elements.blockchain.bsc.provider";

/**
```

```
 * Used to specify the session timeout, in seconds
 */
@ElementDefaultAttribute("172800")
String SESSION_TIMEOUT_SECONDS = "dev.getelements.elements.session.timeout.seconds";

/**
 * Used to specify the mock session timeout.
 */
@ElementDefaultAttribute("3600")
String MOCK_SESSION_TIMEOUT_SECONDS = "dev.getelements.elements.mock.session.timeout.seconds";

/**
 * Used to specify the host for neo blockchain.
 */
@ElementDefaultAttribute("http://127.0.0.1")
String NEO_BLOCKCHAIN_HOST = "dev.getelements.elements.blockchain.neo.host";

/**
 * Used to specify the port for neo blockchain.
 */
@ElementDefaultAttribute("50012")
String NEO_BLOCKCHAIN_PORT = "dev.getelements.elements.blockchain.neo.port";

/**
 * Used to specify the file path for static content.
 */
@ElementDefaultAttribute("content")
String CDN_FILE_DIRECTORY = "dev.getelements.elements.cdnserve.storage.directory";

/**
 * Used to specify the endpoint file path for cloning static content.
 */
@ElementDefaultAttribute("clone")
String CDN_CLONE_ENDPOINT = "dev.getelements.elements.cdnserve.endpoint.clone";

/**
 * Used to specify the endpoint for serving static content.
 */
@ElementDefaultAttribute("serve")
String CDN_SERVE_ENDPOINT = "dev.getelements.elements.cdnserve.endpoint.serve";
```