UserService API/SPI with Impersonation

Table of Contents

This module (`incode-module-userimpersonate-dom`) overrides the default implementation of Apache Isis' `UserService` API. Intended for prototyping and demos only, it allows the logged-in user to be switched dynamically, ie to impersonate some other user.

API

The `UserService` API is defined as:

```
public interface UserService {
    UserMemento getUser();
}
```

where `UserMemento` contains both the current user name and their set of roles.

The Apache Isis framework provide a default implementation of this service. What this module does is to provide a different implementation of the service with a higher priority which is therefore used instead.

This implementation - `UserServiceWithImpersonation` - extends the API as follows:

```
public class UserServiceWithImpersonation implements UserService {

    public UserMemento getUser() { ... }                                ①

    public void setUser(String userName) { ... }                        ②
    public void setUser(String userName, String... roles) { ... }
    public void setUser(String userName, List<String> roles) { ... }

    public void reset() { ... }                                         ③

    public boolean isImpersonating() { ... }                            ④

    public boolean isAvailable() { ... }                                ⑤
}
```

① the API of `UserService`, this returns the current user. If a user is being impersonated (per `setUser(…)`) then that will be returned.

Otherwise, though, the service delegates down to the default implementation of `UserService` (in other words, the user that actually logged on).

② Impersonate a different user, optionally specifying a different set of roles. If roles are not specified then the roles of the original user are preserved.

③ Removes the impersonation, returning back to the original user.

④ Check to see if a user is being impersonated (that is, that `setUser(…)` was called and `reset()` has not been called subsequently).

⑤ Checks that the service is available. If it is not, then `setUser(…)` has no effect.

The service is only available when running within a webapp. This is because it relies upon the `HttpSession` (as obtained from Servlet API module) to store the impersonating user identity.

Note that impersonations do *not* nest.

That is, the user logs in as *user1*, then impersonates to *user2*, then impersonates again to *user3*, then resetting will go back to the original *user1*.

Menus and Contributions

The `ImpersonationMenu` domain service provides a couple of actions which are surfaced in the UI on the tertiary menu bar:

```
public class ImpersonateMenu {

    public void impersonate(
            ApplicationUser applicationUser,
            @Nullable
            List<ApplicationRole> applicationRoleList) { ... }

    public void stopImpersonating() { ... }
}
```

This uses the Security module to constrain the list of valid users/roles.

There are also two mixins:

- the `Object_impersonateUser` mixin allows impersonation to be invoked from any domain object.
- the `Object_stopImpersonating` mixin similarly resets impersonation.

The actions of the menu and the mixins are restricted to prototyping only. They also emit domain events so can be vetoed if required.

Behind the scenes these classes delegate to the module's `ImpersonationService`, mostly an implementation detail:

```
public class ImpersonationService {

    public void impersonate(
            ApplicationUser applicationUser,
            List<ApplicationRole> applicationRoleList) { ... }
    public boolean hideImpersonate() { ... }

    public void stopImpersonating() { ... }
    public boolean hideStopImpersonating() { ... }
}
```

How to configure/use

Classpath

Update your classpath by adding this dependency in your `dom` project's `pom.xml`:

```
<dependency>
    <groupId>org.incode.module.userimpersonate</groupId>
    <artifactId>incode-module-userimpersonate-dom</artifactId>
    <version>1.16.1</version>
</dependency>
```

Check for later releases by searching Maven Central Repo.

For instructions on how to use the latest `-SNAPSHOT`, see the contributors guide.

Bootstrapping

In the `AppManifest`, update its `getDependencies()` method, eg:

```
@Override
public Set<Module> getDependencies() {
    return Sets.newHashSet(
            ...
            new org.incode.module.userimpersonate.UserImpersonateModule(),
            ...
    );
}
```

Known issues

None known at this time.

Dependencies

Maven can report modules dependencies using:

```
mvn dependency:list -o -pl modules/spi/userimpersonate/impl -D excludeTransitive=true
```

which, excluding Apache Isis itself, depends only on the ServletAPI library module and the Security SPI module