**Use Case 03a\_ Summon Hero**

**Explanation.**

1. The user clicks on the Summon Hero button on the MainAction page, an HIC object, which was created earlier when Mainframe was created during initialization.
2. The MainAction’s Summon Hero button’s action method--the anonymous actionPerformed() method--calls the summonHeroes() method on the MainActionCiv object, waiting it to return a list of Hero name plates. MainActionCiv was created when the MainAction page was created.
3. MainActionCiv.summonHeroes() calls the HeroRegistry method getNamePlates(). HeroRegistry is obtained from a call to the RegistryFactory singleton, either at this time, or some time previously, It has existed since initialization.
4. HeroRegistry loops through every Hero asking each Hero to return that Hero’s nameplate. The HeroRegistry put all the nameplates into a name plate list and returns it back to the MainActionCiv, which in turn for display. (The left-side rising arrow shows that the control is a private loop method.)
5. MainAction calls its private method (indicated by a loop) to display the name plate list (e.g., dropdown box).
6. The user selectes the Hero desired from the nameplate list.
7. The drop down list calls MainActionCiv.selectHero() to extract the hero name from the name plage.
8. MainActionCiv calls HeroRegistry.getName(name) to get the Hero requested, which is returned.
9. The HeroDisplay constructor is called by passing the Hero object to it. (Constructor notation is designated by the keyword new.) This is an example of constructor injection.
10. Part of the work of the constructor is to display the Hero object: attributes, inventory, magic items, and if appropriate, magic spells. All the work of HeroDisplay is the same as was done in Create Hero, so no further detail is necessary here.

Behind the scenes, in order to display all the elements of the Hero, the ItemRegistry, OccupationRegistry, SkillRegistry, etc. will need to be opened so those component objects’ data can be retrieved and displated.