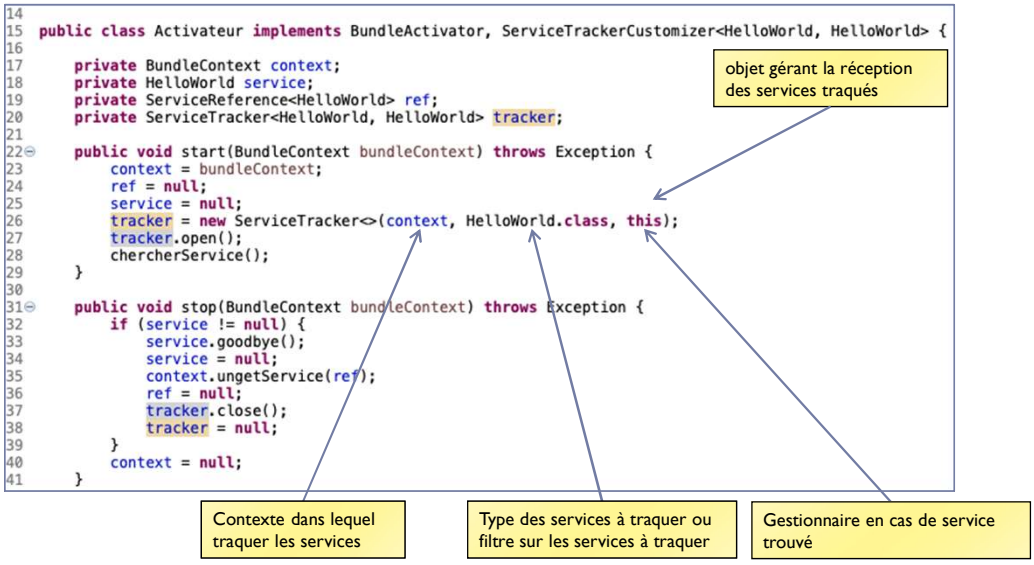
Enregistrement des ServiceTracker

- Pour chaque service requis par un composant (bundle) B
  - On enregistre dans la classe d'activation de B, le ServiceTracker correspondant
  - On ouvre le tracker pour qu'il commence à travailler (à traquer les services)



Dans cette exemple nous avons utilisé la classe Activateur comme classe qui implémente l'interface ServiceTrackerCustomizer<S,T>

On peut aussi hériter de la classe ServiceTracker et faire la même chose.

public class Activateur2 **extends** ServiceTracker<HelloWorld, HelloWorld> **implements** BundleActivator{

public Activateur2(BundleContext context, Class<HelloWorld> clazz, ServiceTrackerCustomizer<HelloWorld, HelloWorld> customizer) {

super(context, clazz, customizer);

}

// etc.

}

Remarque : la méthode chercherService() dans start n'est pas nécessaire. La méthode open le fait suite à l'envoi de'évts

Définition d'un ServiceTrackerCustomizer

- Un ServiceTrackerCustomizer est appelé lorsque le ServiceTracker associé est tenu au courant d'un événement sur un service qu'il traque

```
6  import org.osgi.framework.BundleContext;
7  import org.osgi.framework.InvalidSyntaxException;
8  import org.osgi.framework.ServiceReference;
9  import org.osgi.util.tracker.ServiceTrackerCustomizer;
10
11 import exemple.contrat.HelloWorld;
12
13 public class HelloWorldTraqueurService implements ServiceTrackerCustomizer<HelloWorld, HelloWorld> {
14     private ServiceReference<HelloWorld> ref;
15     private BundleContext context;
16     private HelloWorld service;
17
18     public HelloWorldTraqueurService(BundleContext bc, ServiceReference<HelloWorld> sr, HelloWorld s) {
19         context = bc; ref = sr; service = s;
20     }
21
22     public HelloWorld addingService(ServiceReference<HelloWorld> sr) {
38
39     public void modifiedService(ServiceReference<HelloWorld> sr, HelloWorld service) {
45
46     public void removedService(ServiceReference<HelloWorld> sr, HelloWorld service) {
55
56     private void chercherService() {
74 }
75
```


On peut aussi définir une nouvelle classe pour implémenter l'interface ServiceTrackerCustomizer<S,T>

On peut aussi hériter de la classe ServiceTracker et faire la même chose.

Dans l'activateur, il faut créer l'objet instance de cette classe avant de créer le tracker :

```
public void start(BundleContext bundleContext) throws Exception {
        Activateur2.context = bundleContext;
        ref=null;
        service=null;
        tracker = new ServiceTracker<>(context, HelloWorld.class, new
HelloWorldTraqueurService(context, ref, service));
        tracker.open();
}
```