Event Service

Event Service propagation of events and registration of event subscribers for notification of events.

Example of usage. Imagine we have some class *MyService* that is interested to events of type *MyEvent*. Need create implementation of interface *EventSubscriber<T>* parameterized with *MyEvent* type. Examples of code below assume usage of Guice container (or some other IoC container). *MyEventSubscriber* is registered when container starts and removed when container stops.

```
...
import com.codenvy.api.core.notification.EventService;
...

@Singleton
public class MyService {

    private final EventService eventService;
    private final EventSubscriber<MyEvent> subscriber;

 @Inject
    MyService(EventService eventService) {
   this.eventService = eventService;

  subscriber = new MyEventSubsciber();
 }

 @PostConstruct
    void start() {
  ...
     evetnService.subscribe(subscriber);
  ...
 }

 @PreDestroy
    void stop() {
  ...
     evetnService.unsubscribe(subscriber);
  ...
 }
}
```

```
...
import com.codenvy.api.core.notification.EventSubscriber;
...

public class MyEventSubscriber implements EventSubscriber<MyEvent> {
 @Override
    public void onEvent(MyEvent event) {
     // do something with event
 }
}
```

```
public class MyEvent {
...
}
```

Now *MyEventSubscriber* is notified about publishing of *MyEvent* in *EventService*. *MyEventSubscriber* is also notified about publishing all events that are subclasses of *MyEvent.*

Here is example of publishing event to *EventService:*

```
...
import com.codenvy.api.core.notification.EventService;
...

@Singleton
public class MyEventPublisher {

    private final EventService eventService;

 @Inject
    MyEventPublisher(EventService eventService) {
   this.eventService = eventService;
  }

   public void doSomething() {
  // Do some work and notify when done.
  ...
     evetnService.publish(new MyEvent());
 }
}
```