Distributed EventBus Configuration

See about Event Service

How to share events between two or more servers

In some cases it is necessary share events between different servers. For instance slave runner servers need notify run queue when application is started.

Events are shared over websocket connection. Need configure client and server. Both of them might send and receive events.

Server configuration

- bind server component (with Guice module)

```
@DynaModule
public class MyModule extends AbstractModule {
    @Override
    protected void configure() {
  ...
  bind(com.codenvy.api.core.notification.WSocketEventBusServer.class);
 }
}
```

- add configuration parameter *notification.server.propagate_events*. This parameter describes list of events that server allowed to propagate over websocker connection. In this case events classes MUST be annotated with *com.codenvy.api.core.notification.EventOrigin* annotation, see example bellow. If need to receive events only then set property to **empty string**, e.g. *notification.server.propagate_events=*

```
notification.server.propagate_events=event_origin[,event_origin]
```

Client configuration

- bind client component (with Guice module)

```
@DynaModule
public class MyModule extends AbstractModule {
    @Override
    protected void configure() {
  ...

bind(com.codenvy.api.core.notification.WSocketEventBusClient.class).asEagerSingleton();
 }
}
```

- add configuration parameter *notification.client.propagate_events*. This parameter describes list of events that client is allowed to propagate over websocker connection. In this case events classes MUST be annotated with *com.codenvy.api.core.notification.EventOrigin* annotation, see example bellow. If need to receive events only then set property to **empty string**, e.g. *notification.client.propagate_events=*
- add configuration parameter *notification.client.event_subscriptions*. This parameter describes list of events that client is interested to get over websocker connection. In this case events classes MUST be annotated with *com.codenvy.api.core.notification.EventOrigin* annotation, see example bellow. If need to sends events only then set property to **empty string**, e.g. *notification.client.event_subscriptions=*

```
notification.client.propagate_events=websocket_url=event_origin[,websocket_url=ev
ent_origin]
notification.client.event_subscriptions=websocket_url=event_origin[,websocket_url
=event_origin]
```

Here is example of configuration client and server.

1. First create two classes for events we want to share, class must conform to Java Bean specification.

   ```
   ...
   import com.codenvy.api.core.notification.EventOrigin;
   ...
   @EventOrigin("test_server")
   public class MyServerEvent {
   ...
   }
   ```

   ```
   ...
   import com.codenvy.api.core.notification.EventOrigin;
   ...
   @EventOrigin("test_client")
   public class MyClientEvent {
   ...
   }
   ```

2. Configure server. Bind component.

   ```
   bind(com.codenvy.api.core.notification.WSocketEventBusServer.class);
   ```

   Add configuration property. If need share more then one event add them in comma-separated list.

   ```
   notification.server.propagate_events=test_server
   ```

3. Configure client. Bind component.

   ```
   bind(com.codenvy.api.core.notification.WSocketEventBusClient.class).asEagerSingle
   ton();
   ```

   Add configuration properties.

   ```
   notification.client.event_subscriptions=ws://localhost:8080/api/ws/=test_server
   notification.client.propagate_events=ws://localhost:8080/api/ws/=test_client
   ```

   notification.client.event_subscriptions - says get events "test_server" from server ws://localhost:8080/api/ws/

   notification.client.propagate_events - says send events "test_client" to server ws://localhost:8080/api/ws/
   If need send more than one type of event then add construction above as comma-separated list, e.g.

```
notification.client.event_subscriptions=ws://localhost:8080/api/ws/=test_server,
ws://localhost:8080/api/ws/=test_server2
notification.client.propagate_events=ws://localhost:8080/api/ws/=test_client,
ws://localhost:8080/api/ws/=test_client2
```