70.2 Change the HTTP port

In a standalone application the main HTTP port defaults to `8080`, but can be set with `server.port` (e.g. in `application.properties` or as a System property). Thanks to relaxed binding of `Environment` values you can also use `SERVER_PORT` (e.g. as an OS environment variable).

To switch off the HTTP endpoints completely, but still create a `WebApplicationContext`, use `server.port=-1` (this is sometimes useful for testing).

For more details look at *Section 27.3.4, "Customizing embedded servlet containers"* in the 'Spring Boot features' section, or the `ServerProperties` source code.

70.3 Use a random unassigned HTTP port

To scan for a free port (using OS natives to prevent clashes) use `server.port=0`.

70.4 Discover the HTTP port at runtime

You can access the port the server is running on from log output or from the `EmbeddedWebApplicationContext` via its `EmbeddedServletContainer`. The best way to get that and be sure that it has initialized is to add a `@Bean` of type `ApplicationListener<EmbeddedServletContainerInitializedEvent>` and pull the container out of the event when it is published.

Tests that use `@SpringBootTest(webEnvironment=WebEnvironment.RANDOM_PORT)` can also inject the actual port into a field using the `@LocalServerPort` annotation. For example:

```
@RunWith(SpringJUnit4ClassRunner.class)
@SpringBootTest(webEnvironment=WebEnvironment.RANDOM_PORT)
public class MyWebIntegrationTests {

    @Autowired
    EmbeddedWebApplicationContext server;

    @LocalServerPort
    int port;

    // ...

}
```

`@LocalServerPort` is a meta-annotation for `@Value("${local.server.port}")`. Don't try to inject the port in a

> regular application. As we just saw, the value is only set once the container has initialized; contrary to a test, application code callbacks are processed early (i.e. before the value is actually available).

70.5 Configure SSL