ÖVNING 1 - ROUTER NON-BLOCKING

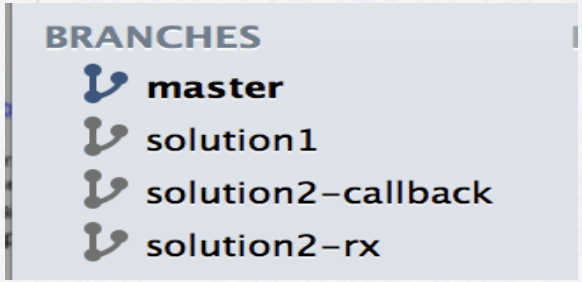
- Ramverk och applikationer
- Router blocking
- Övning 1
  - Starta service-providern
  - Manuella tester av blocking router
  - Lasttester av blocking router
  - Omkodning till non-blocking
  - Tester (manuella och lasttester)

RAMVERK OCH APPLIKATIONER I VÅRA ÖVNINGAR

- Java 8
- GIT
- Gradle
- Spring Boot och Spring MVC
- Logback
- Eclipse

- RealTimeLoadTester (RTLT)

GIT



- Boxen är uppdaterad med det senaste från GitHub (master)
  - Uppdatering av branch via kommadot: git pull
- Lösningar till våra övningar finns i egna brancher enligt ovan.
- Tänk på att dessa lösningar kan skriva över din inlagda kod om du hämtar från en av dessa brancher!
  - Byte av branch via kommandot: git checkout <branch>
- Se http://git-scm.com för mer info.

GRADLE

- Som byggsystem använder vi Gradle.
  - Se https://www.gradle.org/ för mer info

- Följande Gradle kommandon kan vara vara användbara:
  - ./gradlew build -- bygger och kör tester på koden
  - ./gradlew clean -- rensar genererad kod från en build
  - ./gradlew eclipse -- genererar Eclipse projekt
  - ./gradlew bootRun -- startar din Spring Boot applikation

- Av dessa är det i princip endast **bootRun** vi kommer att använda.

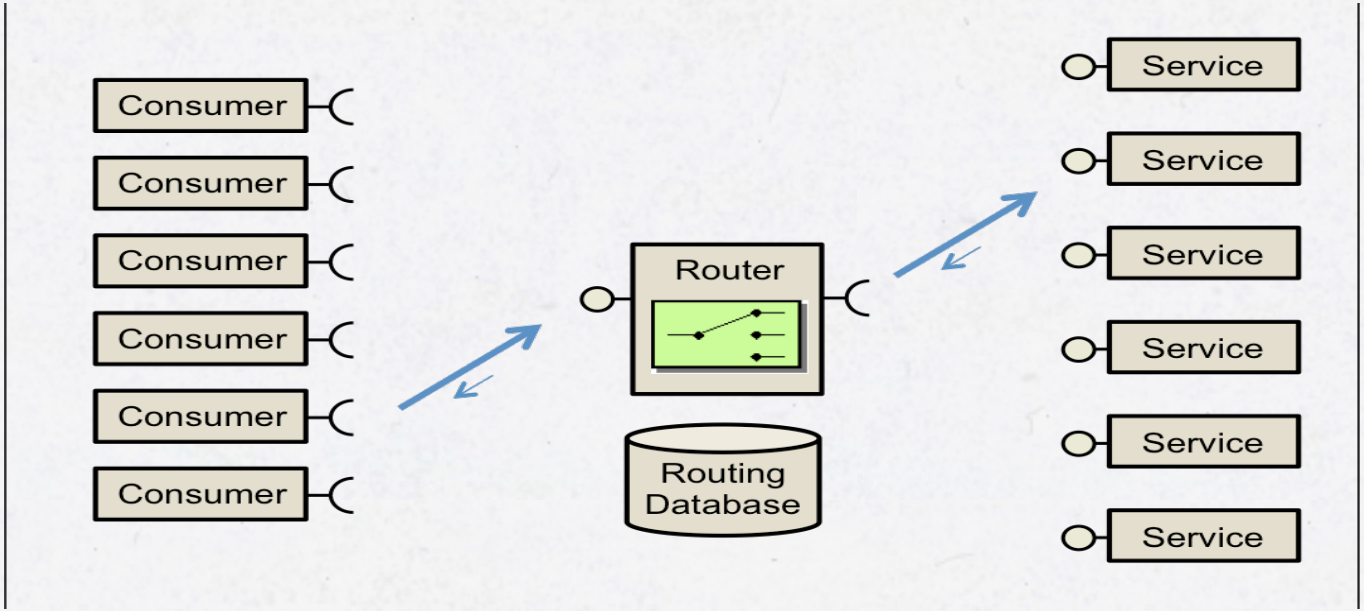
LOGBACK

- Loggkonfiguration – logback.xml
  - Loggfilen läses om automatiskt!

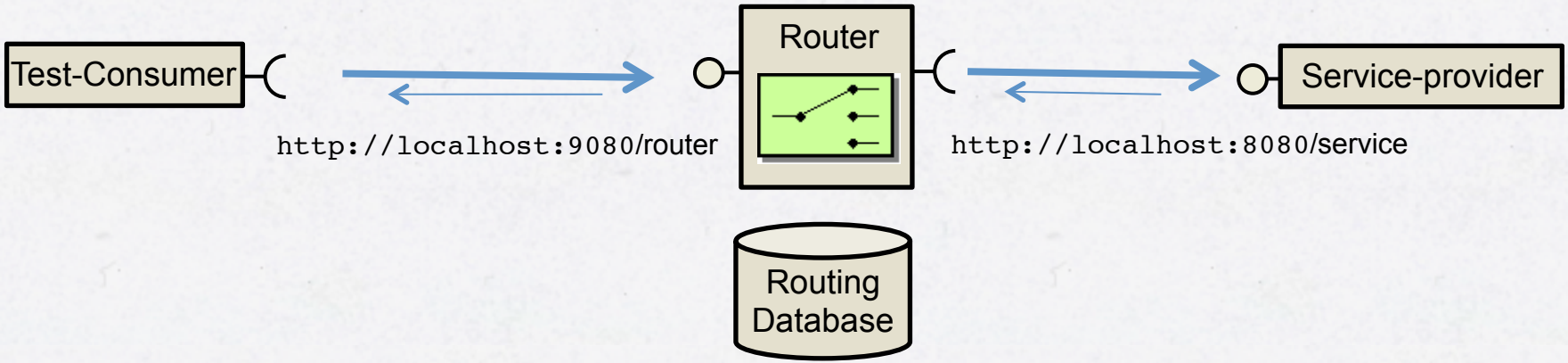
- All loggning sker till konsolen, dvs i kommandofönstret

- Ändra loggnivå
  - Använd DEBUG loggnivån i våra övningar för loggning
  - Byt till INFO för att minska utskrifter i samband med lasttester.

ÖVNING 1 - ROUTER



ÖVNING 1 - ROUTER



SERVICE-PROVIDER

- /service – parameter
  - qry=error – genererar ett fel från tjänsten
  - qry=timeout – genererar en timeout
  - qry=* – normal bearbetning

BLOCKING ROUTER

```
@RestController
public class RouterController {

    @Value("${serviceProvider.url}”) private String url;

    @Autowired private RestTemplate restTemplate;

    @Autowired private UtilBlocking util;

    @RequestMapping("/router")
    public ResponseEntity<String> router(@RequestParam String qry) {

        try {
            return restTemplate.getForEntity(url + "/service?qry=" + qry, String.class);

        } catch (RuntimeException ex) {
            return util.handleException(ex, url);
        }
    }
}
```

ÖVNING 1 TILLVÄGAGÅNGSSÄTT

1. Starta service-providern
2. Manuella tester av blocking router
   - Happy day, felhantering och timeout
3. Lasttester av blocking router
   - Använda RTLT för att driva testerna
4. Skriv non-blocking kod
   - Titta på presentationen
   - Kopiera in halvt ifylld lösning
   - Ersätt ?? Med korrekt kod
   - Ta bort gammal kod
   - Kopiera från färdig lösning vid tidsbrist
5. Tester av non-blocking router
   - Manuella tester och lasttester