Redis

17 Spring Boot Session Store

Session Store

Session

Session Store

HTTP Session

Session Store

Spring Session

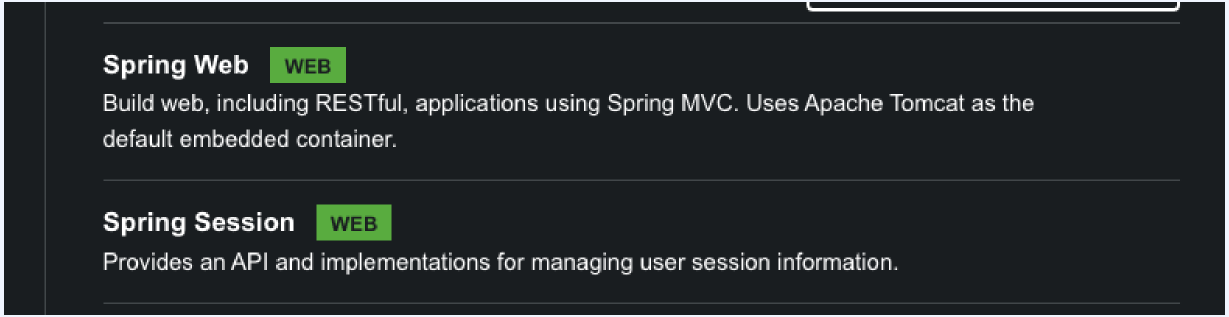
Session Store

Spring Session

1. **HTTP Session**
2. WebSocket
3. WebSession

Session Store

Spring Session



Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

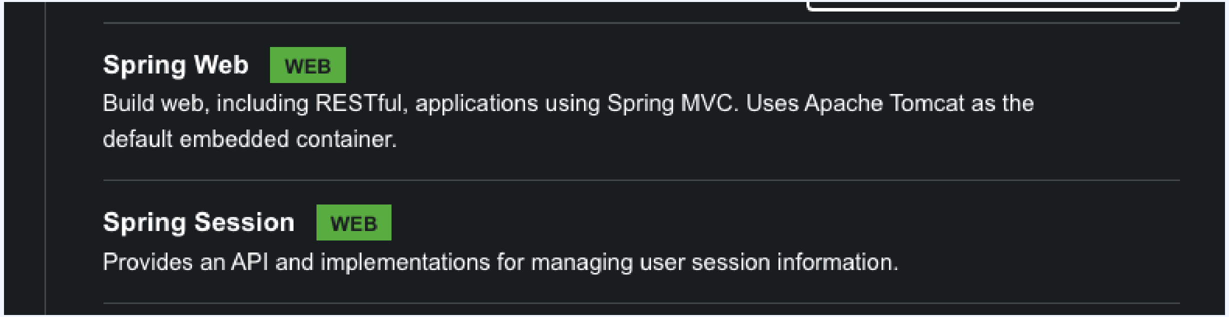
Spring Session WEB

Provides an API and implementations for managing user session information.

https://start.spring.io/

Session Store

Spring Session



Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Session WEB
Provides an API and implementations for managing user session information.

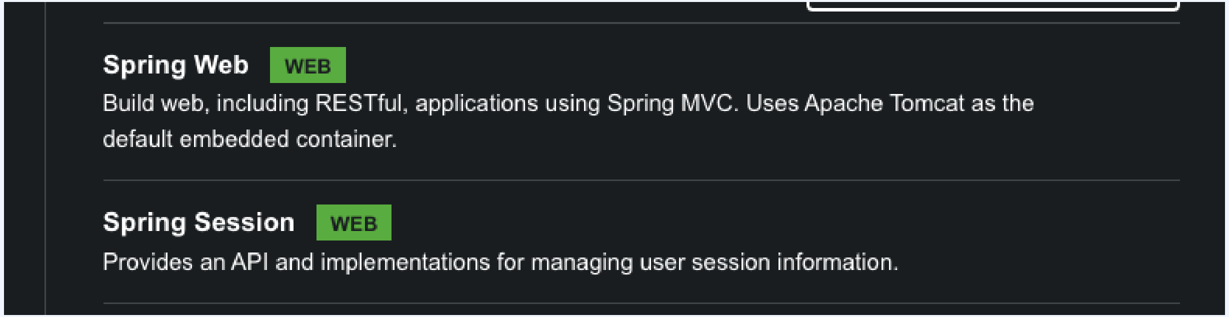
gradle.build

```
implementation 'org.springframework.session:spring-session-core'
```

https://start.spring.io/

Session Store

Spring Session



Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Session WEB

Provides an API and implementations for managing user session information.

gradle.build

~~implementation 'org.springframework.session:spring-session-core'~~

```
implementation 'org.springframework.boot:spring-boot-starter-data-redis'
implementation 'org.springframework.session:spring-session-data-redis'
```

https://start.spring.io/

Session Store

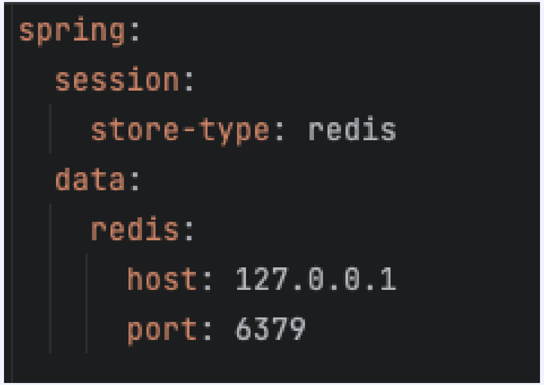
Spring Session

1. **Redis**
2. JDBC
3. Hazelcast
4. Mongodb

Session Store

Spring Session

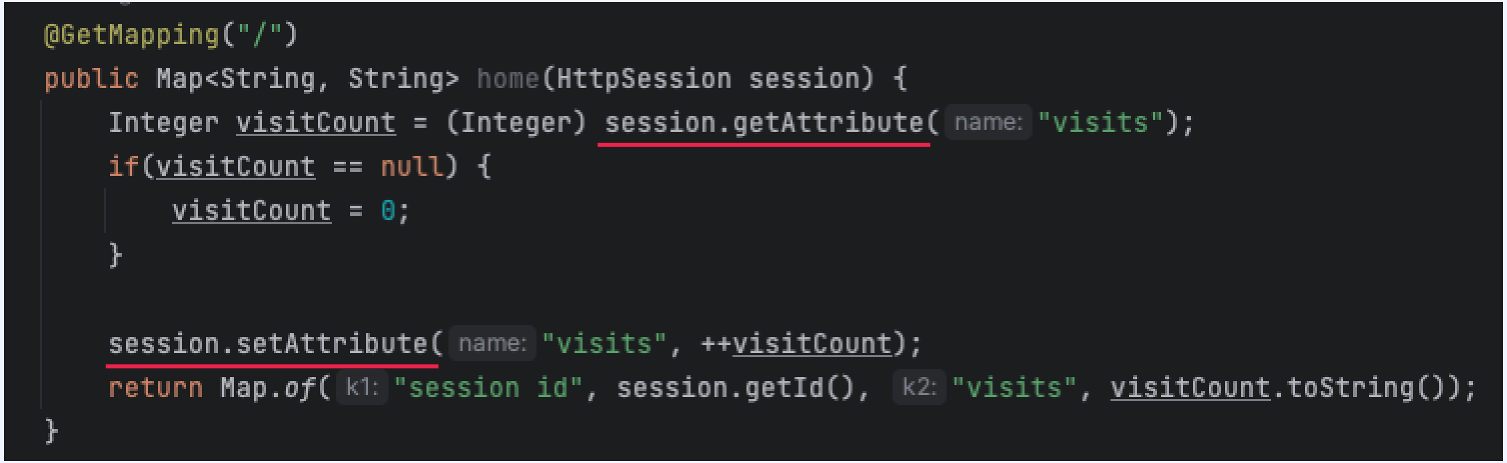
Application.yaml



```
spring:
  session:
    store-type: redis
  data:
    redis:
      host: 127.0.0.1
      port: 6379
```

Session Store

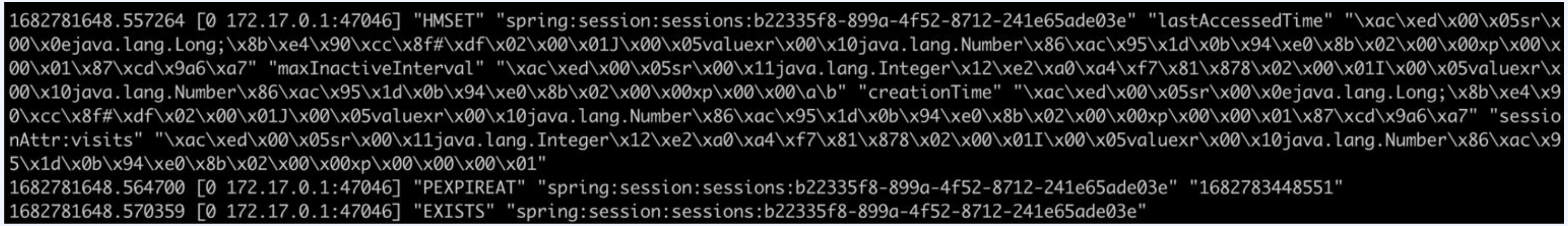
Spring Session



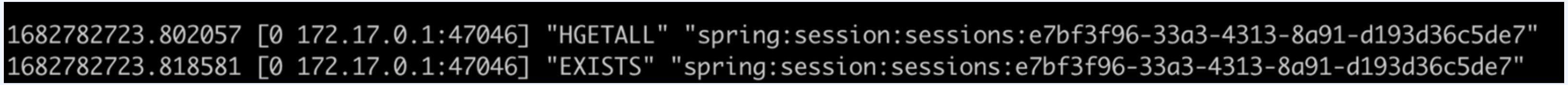
```
@GetMapping("/")
public Map<String, String> home(HttpSession session) {
    Integer visitCount = (Integer) session.getAttribute( name: "visits");
    if(visitCount == null) {
        visitCount = 0;
    }

    session.setAttribute( name: "visits", ++visitCount);
    return Map.of( k1: "session id", session.getId(), k2: "visits", visitCount.toString());
}
```

Session Store

Spring Session



```
1682781648.557264 [0 172.17.0.1:47046] "HMSET" "spring:session:sessions:b22335f8-899a-4f52-8712-241e65ade03e" "lastAccessedTime" "\xac\xed\x00\x05sr\x
00\x0ejava.lang.Long;\x8b\xe4\x90\xcc\x8f#\xdf\x02\x00\x01J\x00\x05valuexr\x00\x10java.lang.Number\x86\xac\x95\x1d\x0b\x94\xe0\x8b\x02\x00\x00xp\x00\x
00\x01\x87\xcd\x9a6\xa7" "maxInactiveInterval" "\xac\xed\x00\x05sr\x00\x11java.lang.Integer\x12\xe2\xa0\xa4\xf7\x81\x878\x02\x00\x01I\x00\x05valuexr\x
00\x10java.lang.Number\x86\xac\x95\x1d\x0b\x94\xe0\x8b\x02\x00\x00xp\x00\x00\a\b" "creationTime" "\xac\xed\x00\x05sr\x00\x0ejava.lang.Long;\x8b\xe4\x9
0\xcc\x8f#\xdf\x02\x00\x01J\x00\x05valuexr\x00\x10java.lang.Number\x86\xac\x95\x1d\x0b\x94\xe0\x8b\x02\x00\x00xp\x00\x00\x01\x87\xcd\x9a6\xa7" "sessio
nAttr:visits" "\xac\xed\x00\x05sr\x00\x11java.lang.Integer\x12\xe2\xa0\xa4\xf7\x81\x878\x02\x00\x01I\x00\x05valuexr\x00\x10java.lang.Number\x86\xac\x9
5\x1d\x0b\x94\xe0\x8b\x02\x00\x00xp\x00\x00\x00\x01"
1682781648.564700 [0 172.17.0.1:47046] "PEXPIREAT" "spring:session:sessions:b22335f8-899a-4f52-8712-241e65ade03e" "1682783448551"
1682781648.570359 [0 172.17.0.1:47046] "EXISTS" "spring:session:sessions:b22335f8-899a-4f52-8712-241e65ade03e"
```



```
1682782723.802057 [0 172.17.0.1:47046] "HGETALL" "spring:session:sessions:e7bf3f96-33a3-4313-8a91-d193d36c5de7"
1682782723.818581 [0 172.17.0.1:47046] "EXISTS" "spring:session:sessions:e7bf3f96-33a3-4313-8a91-d193d36c5de7"
```

Spring Store

실습