The LockManager App Case Study: Server Structure & Functionality (Part 1)

Douglas C. Schmidt

d.schmidt@vanderbilt.edu

www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science

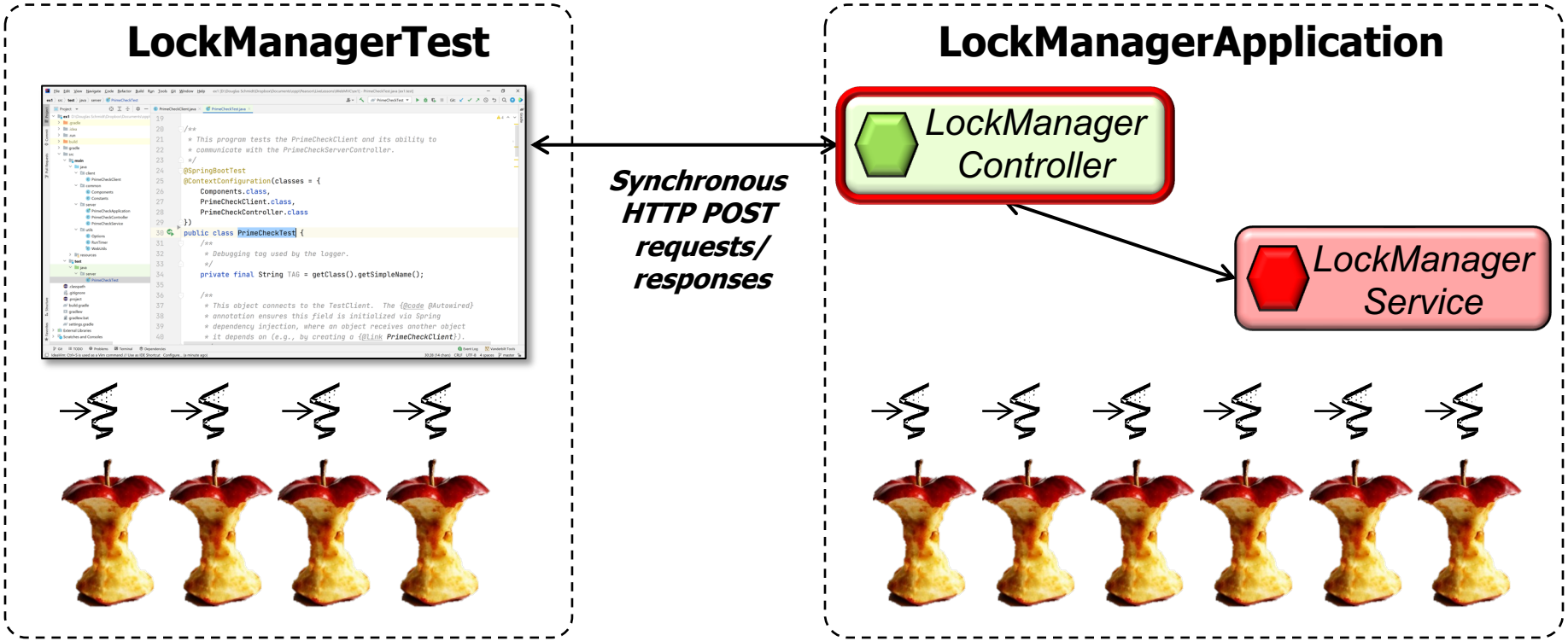
Institute for Software Integrated Systems

Vanderbilt University

Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

- This lesson shows the structure & functionality of the controller class for the LockManager microservice developed using Spring WebMVC



See WebMVC/ex5/src/main/java/edu/vandy/lockmanager/server

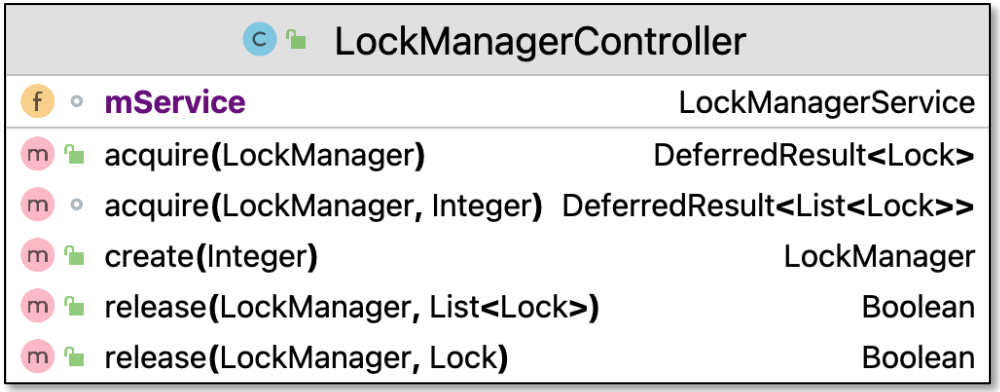
Structure & Functionality of the LockManagerController

Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
   @Autowired
   LockManagerService mService;

   ...
```



See WebMVC/ex5/src/main/java/edu/vandy/lockmanager/server/LockManagerController.java

Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
   @Autowired
   LockManagerService mService;

   ...
```

This annotation ensures request handling methods in the controller class automatically serialize return objects into HttpResponse objects

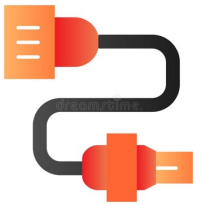
See www.baeldung.com/spring-controller-vs-restcontroller

Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
   @Autowired
   LockManagerService mService;

   ...
```



This field is auto-wired by Spring's dependency injection framework

See www.baeldung.com/spring-autowire

Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
   ...
   @PostMapping(CREATE)
   public LockManager create(@RequestParam Integer permits)
   ...
   @PostMapping(ACQUIRE_LOCK)
   public DeferredResult<Lock> acquire
            (@RequestParam LockManager lockManager) ...

   @PostMapping(RELEASE_LOCK)
   public Boolean release
            (@RequestParam LockManager lockManager,
             @RequestBody Lock lock) ...
```

These endpoint handler methods forward to the LockManagerService methods that fulfill the requests

Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
   ...
   @PostMapping(CREATE)
   public LockManager create(@RequestParam Integer permits)
   ...
   @PostMapping(ACQUIRE_LOCK)
   public DeferredResult<Lock> acquire()
            (@RequestParam LockManager lockManager) ...

   @PostMapping(RELEASE_LOCK)
   public Boolean release
            (@RequestParam LockManager lockManager,
             @RequestBody Lock lock) ...
```

These annotations map HTTP POST requests to endpoint handler methods

See www.baeldung.com/spring-new-requestmapping-shortcuts

Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
   ...
   @PostMapping(CREATE)
   public LockManager create(@RequestParam Integer permits)
   ...
   @PostMapping(ACQUIRE_LOCK)
   public DeferredResult<Lock> acquire()
            (@RequestParam LockManager lockMan

   @PostMapping(RELEASE_LOCK)
   public Boolean release
            (@RequestParam LockManager lockMan
             @RequestBody Lock lock) ...
```

HTTP POST requests are used since server state is modified

See www.baeldung.com/cs/http-get-vs-post

Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
   ...
   @PostMapping(CREATE)
   public LockManager create(@RequestParam
   ...
   @PostMapping(ACQUIRE_LOCK)
   public DeferredResult<Lock> acquire()
            (@RequestParam LockManager lockManager) ...

   @PostMapping(RELEASE_LOCK)
   public Boolean release
            (@RequestParam LockManager lockManager,
             @RequestBody Lock lock) ...
```

These strings identify & route to endpoint handler methods from incoming HTTP POST requests

See www.baeldung.com/spring-new-requestmapping-shortcuts

Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
   ...
   @PostMapping(CREATE)
   public LockManager create(@RequestParam Integer permits)
   ...
   @PostMapping(ACQUIRE_LOCK)
   public DeferredResult<Lock> acquire()
            (@RequestParam LockManager lockManager) ...

   @PostMapping(RELEASE_LOCK)
   public Boolean release
            (@RequestParam LockManager lockManager,
             @RequestBody Lock lock) ...
```

This annotation maps the one-&-only Http Request body to a Java object

See www.baeldung.com/spring-request-response-body

Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
   ...
   @PostMapping(CREATE)
   public LockManager create(@RequestParam Integer permits)
   ...
   @PostMapping(ACQUIRE_LOCK)
   public DeferredResult<Lock> acquire()
            (@RequestParam LockManager lockManager) ...

   @PostMapping(RELEASE_LOCK)
   public Boolean release
            (@RequestParam LockManager
             @RequestBody Lock lock) ...
```

This annotation can extract query parameters, form params, & files from an HTTP request

See www.baeldung.com/spring-request-param

Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
   ...
   @PostMapping(CREATE)
   public LockManager create(@RequestParam
   ...
   @PostMapping(ACQUIRE_LOCK)
   public DeferredResult<Lock> acquire()
            (@RequestParam LockManager lockManager) ...

   @PostMapping(RELEASE_LOCK)
   public Boolean release
            (@RequestParam LockManager lockManager,
             @RequestBody Lock lock) ...
```

This Java class enables the microservice to generate the result via a background thread the service choses

See springframework/web/context/request/async/DeferredResult.html

Structure & Functionality of the LockManagerController

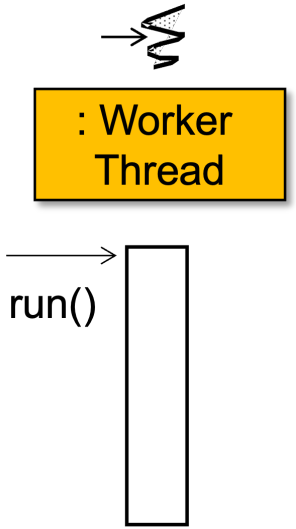
- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
  @PostMapping(ACQUIRE_LOCK)
  public DeferredResult<Lock> acquire()
           (@RequestParam LockManager lockManager) {
    var result = new DeferredResult<Lock>();

    var callback = new Callback() {
      public void onSuccess(Lock lock)
      {result.setResult(lock);}
      public void onError(Throwable error)
      { result.setErrorResult(error); }};

    mService.acquire(lockManager, callback);
    return result; ...
```

Acquire a Lock via Spring's DeferredResult mechanism



The LockManagerController acquire() method runs in an HTTP worker thread

Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
  @PostMapping(ACQUIRE_LOCK)
  public DeferredResult<Lock> acquire()
            (@RequestParam LockManager lockManager) {
    var result = new DeferredResult<Lock>();

    var callback = new Callback() {
      public void onSuccess(Lock lock)
      {result.setResult(lock);}
      public void onError(Throwable error)
      { result.setErrorResult(error); }};

    mService.acquire(lockManager, callback);
    return result; ...
```

Helps offload a long-running operation from an HTTP worker thread to a (virtual) thread

See springframework/web/context/request/async/DeferredResult.html

Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
  @PostMapping(ACQUIRE_LOCK)
  public DeferredResult<Lock> acquire()
            (@RequestParam LockManager lockManager) {
    var result = new DeferredResult<Lock>();

    var callback = new Callback() {
      public void onSuccess(Lock lock)
      {result.setResult(lock);}
      public void onError(Throwable error)
      { result.setErrorResult(error); }};

    mService.acquire(lockManager, callback);
    return result; ...
```

Handles completion or failure of an asynchronous operation

See WebMVC/ex5/src/main/java/edu/vandy/lockmanager/common/Callback.java

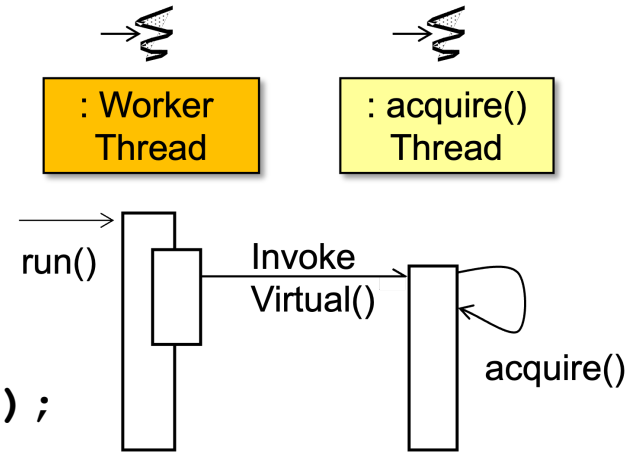
Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
  @PostMapping(ACQUIRE_LOCK)
  public DeferredResult<Lock> acquire()
            (@RequestParam LockManager lockManager) {
    var result = new DeferredResult<Lock>();

    var callback = new Callback() {
      public void onSuccess(Lock lock)
      {result.setResult(lock);}
      public void onError(Throwable error)
      { result.setErrorResult(error); }};

    mService.acquire(lockManager, callback);
    return result; ...
```



Run acquire() & associated callback in a virtual thread

Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
  @PostMapping(ACQUIRE_LOCK)
  public DeferredResult<Lock> acquire()
            (@RequestParam LockManager lockManager) {
    var result = new DeferredResult<Lock>();

    var callback = new Callback() {
      public void onSuccess(Lock lock)
      {result.setResult(lock);}
      public void onError(Throwable error)
      { result.setErrorResult(error); }};

    mService.acquire(lockManager, callback);
    return result; ...
```

Return the Deferred Result immediately

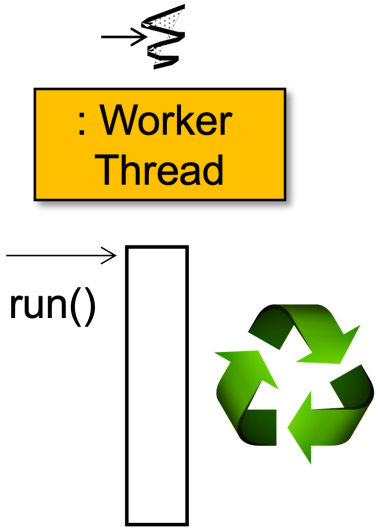
Structure & Functionality of the LockManagerController

- LockManagerController maps HTTP POST requests to endpoint handlers

```
@RestController
public class LockManagerController {
  @PostMapping(ACQUIRE_LOCK)
  public DeferredResult<Lock> acquire()
          (@RequestParam LockManager lockManager) {
    var result = new DeferredResult<Lock>();

    var callback = new Callback() {
      public void onSuccess(Lock lock)
      {result.setResult(lock);}
      public void onError(Throwable error)
      { result.setErrorResult(error); }};

    mService.acquire(lockManager, callback);
    return result; ...
```



The HTTP worker thread can be recycled after acquire() returns

End of the LockManager App Case Study: Server Structure & Functionality (Part 1)