Automatically writing Java unit tests with Diffblue Cover

A Tutorial

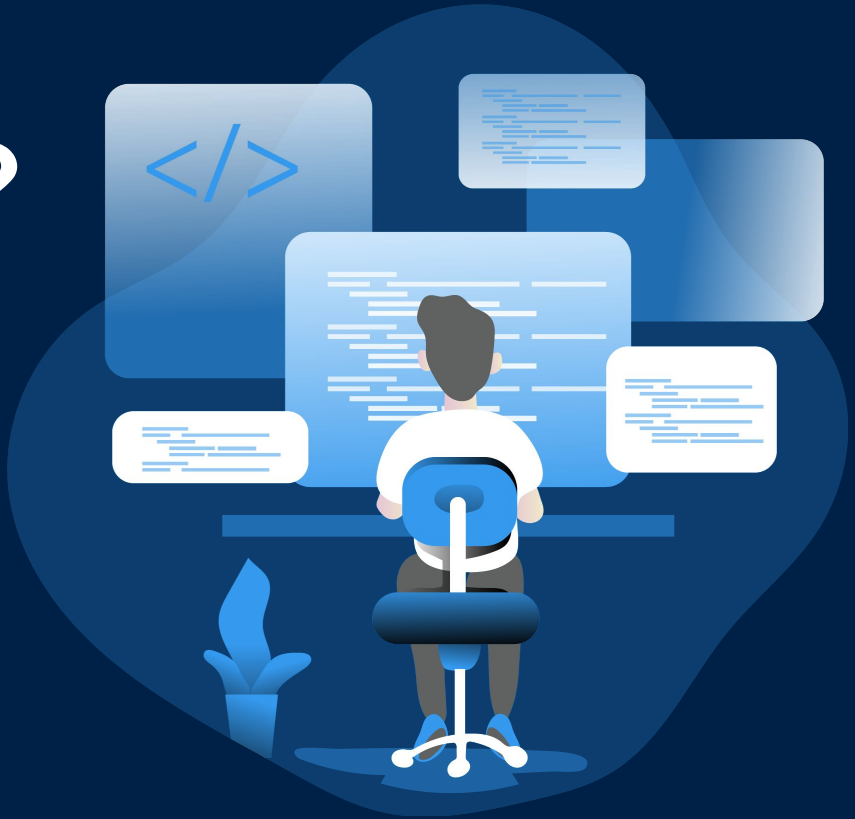
Peter Schrammel

diffblue
AI for Code

Outline

- Diffblue Cover: what it is and how it works
- Using Diffblue Cover
  - when refactoring code
  - when implementing new features
  - together with TDD
- Q&A

What is Diffblue Cover?



IntelliJ
> Preferences
> Plugins
> Marketplace
> Search for “Diffblue Cover”
> Install Community Edition

https://plugins.jetbrains.com/plugin/14946-diffblue-cover

diffblue
AI for Code

What is Diffblue Cover?

1. git clone
   https://github.com/Diffblue-benchmarks/Spring-petclinic
2. cd Spring-petclinic
3. git checkout a-test-intro
4. Open project in IntellIJ
5. Right-click on `CloudStorageService` > Write tests

```
@Service
public class CloudStorageService {

  @Autowired
  private AmazonS3 s3client;

  public PutObjectResult uploadFileToBucket(String bucketName, String key, File file) {
     return s3client.putObject(bucketName, key, file);
  }

  public S3Object downloadFileFromBucket(String bucketName, String key) {
     return s3client.getObject(bucketName, key);
  }

}
```

diffblue
AI for Code

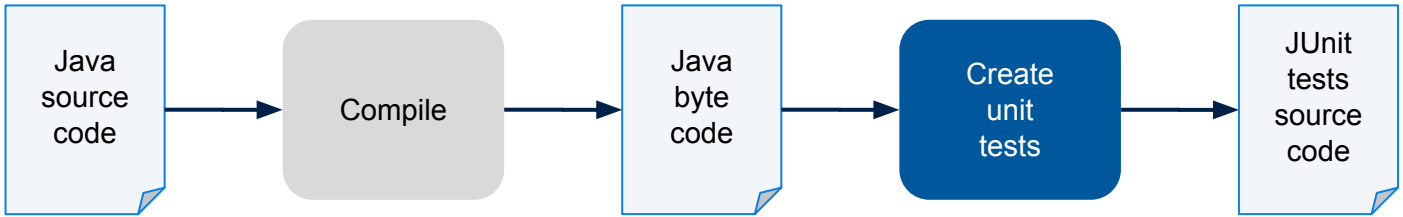
What is Diffblue Cover?

1. git clone
   https://github.com/Diffblue-benchmarks/Spring-petclinic
2. cd Spring-petclinic
3. git checkout a-test-intro
4. Open project in IntellIJ
5. Right-click on `CloudStorageService` > Write tests

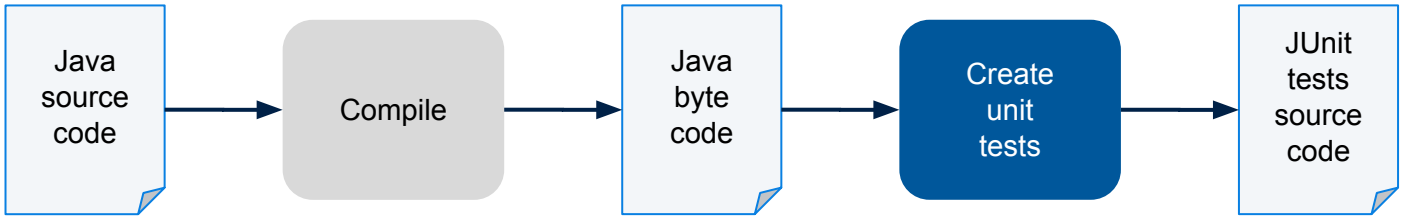
```
@ExtendWith(SpringExtension.class)
public class CloudStorageServiceTest {
  @MockBean
  private AmazonS3Client amazonS3Client;

  @Autowired
  private CloudStorageService cloudStorageService;

  @Test
  public void testDownloadFileFromBucket() throws UnsupportedEncodingException {
     StringInputStream objectContent = new StringInputStream("Lorem ipsum dolor sit amet.");
     S3Object s3Object = new S3Object();
     s3Object.setObjectContent(objectContent);
     when(this.amazonS3Client
        .getObject(or(isA(String.class), isNull()), or(isA(String.class), isNull())))
        .thenReturn(s3Object);
     assertSame(s3Object, this.cloudStorageService.downloadFileFromBucket("bucket-name", "key"));
  }
....
```

diffblue
AI for Code

What does it do?

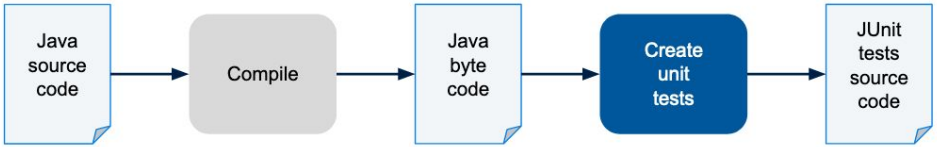


What does it do?



- Tries to trigger execution of paths through a method
- Tries to use inputs that are similar to what a developer would choose
- Mocks dependencies that should be mocked
- Figures out what useful assertions may be
- Assertions always reflect current behavior

What does it do?



- Tries to trigger execution of paths through a method
- Tries to use inputs that are similar to what a developer would choose
- Mocks dependencies that should be mocked
- Figures out what useful assertions may be
- Assertions always reflect current behavior

What it doesn’t do…

- It doesn’t guess the intended behavior.
- It doesn’t replace all manual test writing.
- Does not create end-to-end (e.g. UI) tests.

diffblue
AI for Code

How should I use it?

- Unit regression testing
- Speeding up unit test writing
- Finding bugs by review

Use cases we'll look into:

- I'm going to do a refactoring
- I'm implementing a new feature
- I'm using TDD

Currently supports

- Ubuntu 18-20, macOS 10.15, Windows 10
- IntelliJ CE/Ultimate 2019.3+
- Java 8 and 11
- JUnit 4.7+ and 5
- Mockito 2.1+
- Spring 5+ / Spring Boot 2+

https://docs.diffblue.com

diffblue
AI for Code

What is a unit test?

Desirable properties:

- Runs fast (a few ms)
- Has no side effects on other tests

```
@MockBean
private AmazonS3Client amazonS3Client;
@Test
public void testDownloadFileFromBucket() throws UnsupportedEncodingException {
    StringInputStream objectContent = new StringInputStream("Lorem ipsum dolor sit amet.");
    S3Object s3Object = new S3Object();
    s3Object.setObjectContent(objectContent);
    when(this.amazonS3Client
        .getObject(or(isA(String.class), isNull()), or(isA(String.class), isNull())))
        .thenReturn(s3Object);
    S3Object actualS3Object = this.cloudStorageService.downloadFileFromBucket("bucket-name", "key")
    assertSame(s3Object, actualS3Object);
}
```

Arrange inputs and mocks

diffblue
AI for Code

What is a unit test?

Desirable properties:

- Runs fast (a few ms)
- Has no side effects on other tests

```
@MockBean
private AmazonS3Client amazonS3Client;
@Test
public void testDownloadFileFromBucket() throws UnsupportedEncodingException {
    StringInputStream objectContent = new StringInputStream("Lorem ipsum dolor sit amet.");
    S3Object s3Object = new S3Object();
    s3Object.setObjectContent(objectContent);
    when(this.amazonS3Client
        .getObject(or(isA(String.class), isNull()), or(i
        .thenReturn(s3Object);
    S3Object actualS3Object = this.cloudStorageService.downloadFileFromBucket("bucket-name", "key")
    assertSame(s3Object, actualS3Object);
}
```

Arrange inputs and mocks

Act: run method under test

diffblue
AI for Code

What is a unit test?

Desirable properties:

- Runs fast (a few ms)
- Has no side effects on other tests

```
@MockBean
private AmazonS3Client amazonS3Client;
@Test
public void testDownloadFileFromBucket() throws UnsupportedEncodingException {
    StringInputStream objectContent = new StringInputStream("Lorem ipsum dolor sit amet.");
    S3Object s3Object = new S3Object();
    s3Object.setObjectContent(objectContent);
    when(this.amazonS3Client
        .getObject(or(isA(String.class), isNull()), or(i
        .thenReturn(s3Object);
    S3Object actualS3Object = this.cloudStorageService.                    et-name", "key")
    assertSame(s3Object, actualS3Object);
}
```

Arrange inputs and mocks

Act: run method under test

Assert effects

diffblue
AI for Code

Use case: Refactoring

diffblue
AI for Code

Refactoring

1. git checkout a-test-refactoring
2. Right-click on `PetTypeFormatter` > Write tests

```
@Component
public class PetTypeFormatter implements Formatter<PetType> {
  @Autowired
  private PetRepository pets;
...
  @Override
  public PetType parse(String text, Locale locale) throws ParseException {

     Collection<PetType> findPetTypes = this.pets.findPetTypes();
     for (PetType type : findPetTypes) {
        if (type.getName().equals(text)) {
           return type;
        }
     }
     throw new ParseException("type not found: " + text, 0);
  }
}
```

diffblue
AI for Code

Refactoring

1. git checkout a-test-refactoring
2. Right-click on `PetTypeFormatter` > Write tests
3. Run tests with coverage

```
@ContextConfiguration(classes = {PetTypeFormatter.class, PetRepository.class})
@ExtendWith(SpringExtension.class)
public class PetTypeFormatterTest {
@MockBean(name = "petRepository")
private PetRepository petRepository;
@Autowired
private PetTypeFormatter petTypeFormatter;
…
@Test
public void testParse2() throws ParseException {
  PetType petType = new PetType();
  petType.setId(1);
  petType.setName("Dog");
  ArrayList<PetType> petTypeList =new ArrayList<PetType>();
  petTypeList.add(petType);
  when(this.petRepository.findPetTypes()).thenReturn(petTypeList);
  assertSame(petType, this.petTypeFormatter.parse("Dog", new Locale("en")));
}
…
}
```

diffblue
AI for Code

Refactoring

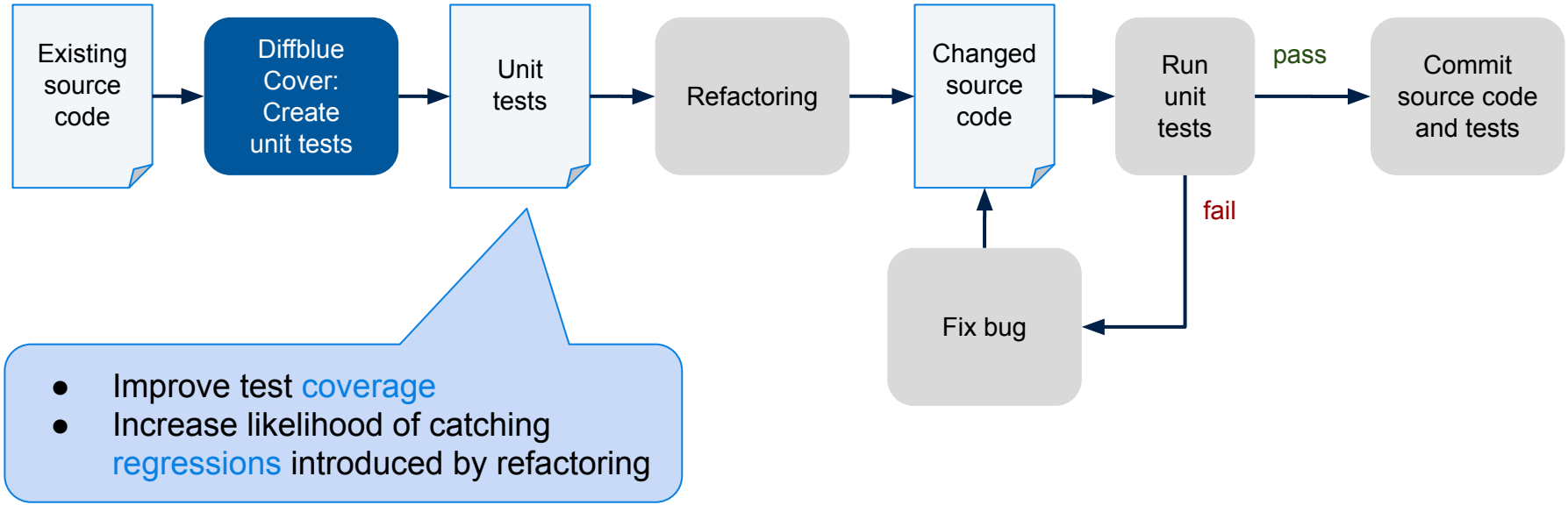
1. git checkout a-test-refactoring
2. Right-click on `PetTypeFormatter` > Write tests
3. Run tests with coverage
4. Refactor `PetTypeFormatter.parse`
5. Run tests (a test fails)
6. Debug failing test
7. Fix bug
8. Run tests (all tests pass)

```
@Component
public class PetTypeFormatter i
  @Autowired
  private PetRepository pets;
...
  @Override
  public PetType parse(String text, Locale locale) throws ParseException {

      return this.pets.findPetTypes().stream()
          .filter(type -> type.getId().equals(text))
          .findFirst()
          .orElseThrow(() -> new ParseException("type not found: " + text, 0));
  }

}
```

Refactoring introduces regression.

diffblue
AI for Code

Refactoring



diffblue
AI for Code

Use case: New Feature

diffblue
AI for Code

New Feature

1. git checkout a-test-new-feature
2. Implement `countDogs` in `PetTypeFormatter`
3. Right-click on `countDogs` > Write tests

```
class OwnerController {
    …
    /**
     * Counts the number of dogs belonging to the given owner.
     * @param owner The owner
     * @return The number of pets of type "Dog" belonging to the given owner
     */
   public long countDogs(Owner owner) {
      return owner.getPets().stream()
         .filter(pet -> pet.getName().equals("Dog"))
         .collect(Collectors.counting());
   }
}
```

diffblue
AI for Code

New Feature

1. git checkout a-test-new-feature
2. Implement `countDogs` in `OwnerController`
3. Right-click on `countDogs` > Write tests
4. Review the created test

```
@Test
public void testCountDogs3() {
  Pet pet = new Pet();
  LocalDate birthDate = LocalDate.ofEpochDay(1L);
  pet.setBirthDate(birthDate);

  Owner owner = new Owner();
  owner.setLastName("Doe");
  owner.setId(1);
  owner.setCity("Oxford");
  owner.setAddress("42 Main St");
  owner.setFirstName("Jane");
  owner.setTelephone("4105551212");
  pet.setOwner(owner);

  pet.setId(1);
  pet.setName("Bella");

  PetType petType = new PetType();
  petType.setId(1);
  petType.setName("Dog");
  pet.setType(petType);

  HashSet<Pet> petSet = new HashSet<Pet>();
  petSet.add(pet);
  owner.setPetsInternal(petSet);

  assertEquals(0L, this.ownerController.countDogs(owner));
}
```

Unintended behavior

diffblue
AI for Code

New Feature

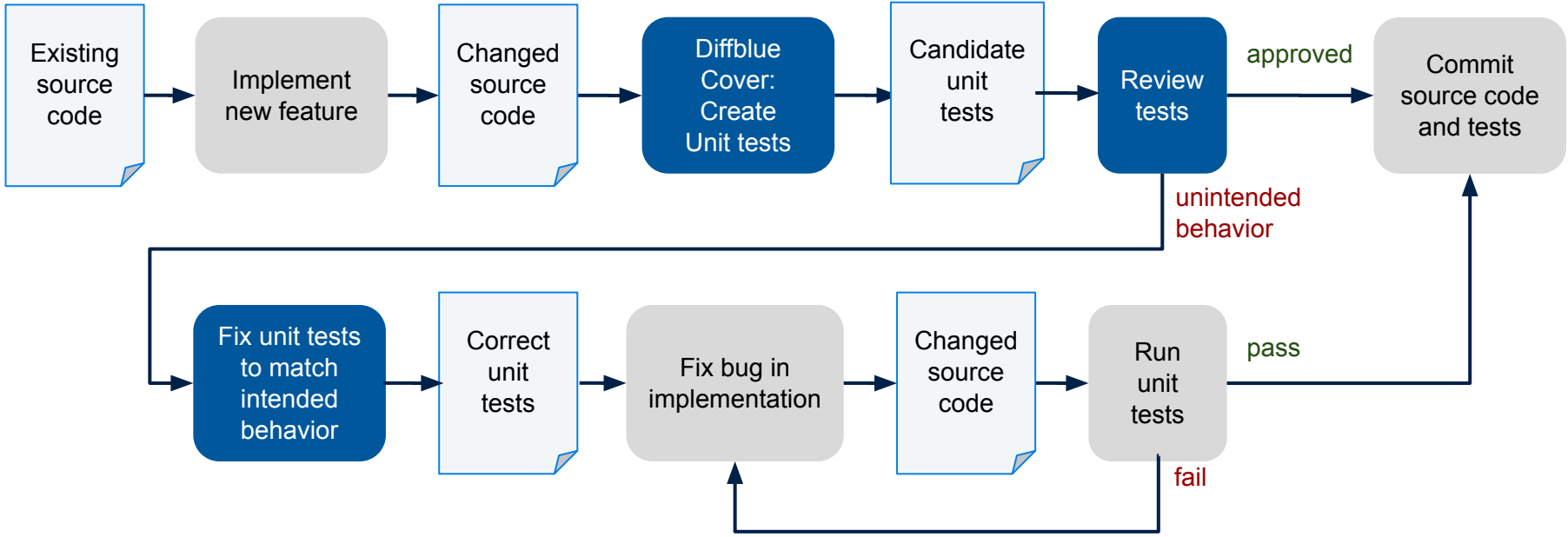
```
@Test
public void testCountDogs3() {
  Pet pet = new Pet();
  LocalDate birthDate = LocalDate.ofEpoch
  pet.setBirthDate(birthDate);

  Owner owner = new Owner();
  owner.setLastName("Doe");
  owner.setId(1);
  owner.setCity("Oxford");
  owner.setAddress("42 Main St");
  owner.setFirstName("Jane");
  owner.setTelephone("4105551212");
  pet.setOwner(owner);

  pet.setId(1);
  pet.setName("Bella");

  PetType petType = new PetType();
  petType.setId(1);
  petType.setName("Dog");
  pet.setType(petType);

  HashSet<Pet> petSet = new HashSet<Pet>();
  petSet.add(pet);
  owner.setPetsInternal(petSet);

  assertEquals(1L, this.ownerController.countDogs(owner));
}
```

1. git checkout a-test-new-feature
2. Implement `countDogs` in `OwnerController`
3. Right-click on `countDogs` > Write tests
4. Review the created tests
5. Fix tests
6. Debug and fix implementation
7. Run tests (pass)

diffblue
AI for Code

New Feature

- Speed up test writing
- Demonstrate unintended behavior in the implementation



diffblue
AI for Code

Use case: Use in TDD

diffblue
AI for Code

TDD

1. git checkout a-test-tdd
2. Implement unit tests for `OwnerController.processFindForm`

```
/**
 * @GetMapping("/owners") public String processFindForm(Owner owner, ...)
 * Look up the owner in the database by the given last name.
 * If a single owner is found, redirect to /owners/{ownerId}.
 * If several owners are found, allow selection of an owner in owners/ownersList.
 */
```

```
@WebMvcTest(controllers = {OwnerController.class})
@ExtendWith(SpringExtension.class)
public class OwnerControllerTest {

  @Autowired
  private MockMvc mockMvc;

  @MockBean(name = "ownerRepository")
  private OwnerRepository ownerRepository;

  @MockBean(name = "visitRepository")
  private VisitRepository visitRepository;
...
```

diffblue
AI for Code

TDD

1. git checkout a-test-tdd
2. Implement unit tests for `OwnerController.processFindForm`
3. Run unit tests (fail)

```
   …
@Test
public void testProcessFindForm_singleOwner() throws Exception {
  Owner owner = new Owner();
  owner.setLastName("Doe");
  owner.setId(1);
  owner.setCity("Oxford");
  owner.setAddress("42 Main St");
  owner.setFirstName("Jane");
  owner.setTelephone("4105551212");
  when(this.ownerRepository.findByLastName(or(isA(String.class), isNull())))
     .thenReturn(Collections.singletonList(owner));

  this.mockMvc.perform(get("/owners").param("lastName", "Doe"))
     .andExpect(status().is3xxRedirection())
     .andExpect(view().name("redirect:/owners/1"));
}
…
```

TDD

1. git checkout a-test-tdd
2. Implement unit tests for `OwnerController.processFindForm`
3. Run unit tests (fail)

```
...
@Test
public void testProcessFindForm_severalOwners()  throws Exception {
  Owner owner1 = new Owner();
  owner1.setLastName("Doe");
  owner1.setId(1);
  owner1.setCity("Oxford");
  owner1.setAddress("42 Main St");
  owner1.setFirstName("Jane");
  owner1.setTelephone("4105551212");

  Owner owner2 = new Owner();
  owner2.setLastName("Doe");
  owner2.setId(1);
  owner2.setCity("Oxford");
  owner2.setAddress("1 High St");
  owner2.setFirstName("John");
  owner2.setTelephone("5121241055");

  when(this.ownerRepository.findByLastName(or(isA(String.class), isNull())))
     .thenReturn(Arrays.asList(owner1, owner2));

  this.mockMvc.perform(get("/owners").param("lastName", "Doe"))
     .andExpect(status().isOk())
     .andExpect(view().name("owners/ownersList"));
}
...
```

diffblue
AI for Code

TDD

1. git checkout a-test-tdd
2. Implement unit tests for `OwnerController.processFindForm`
3. Run unit tests (fail)
4. Implement `processFindForm`
5. Run unit tests (pass)
6. Right-click on `processFindForm` > Write tests

```
public class OwnerController {
…

@GetMapping("/owners")
public String processFindForm(
    Owner owner, BindingResult result, Map<String, Object> model) {

  Collection<Owner> results = this.owners.findByLastName(owner.getLastName());
  if (results.size() == 1) {
     owner = results.iterator().next();
     return "redirect:/owners/" + owner.getId();
  } else {
     model.put("selections", results);
     return "owners/ownersList";
  }
}
```

diffblue
AI for Code

TDD

1. git checkout a-test-tdd
2. Implement unit tests for `OwnerController.processFindForm`
3. Run unit tests (fail)
4. Implement `processFindForm`
5. Run unit tests (pass)
6. Right-click on `processFindForm` > Write tests
7. Review created tests

```
public class OwnerControllerTes

…

@Test
public void testProcessFindForm() throws Exception {

  when(this.ownerRepository.findByLastName(or(isA(String.class), isNull())))
      .thenReturn(new ArrayList<Owner>());

  this.mockMvc.perform(get("/owners"))
      .andExpect(status().isOk());
      .andExpect(view().name("owners/ownersList"));
}
```

What's the intended behavior in this case?

diffblue
AI for Code

TDD

1. git checkout a-test-tdd
2. Implement unit tests for `OwnerController.processFindForm`
3. Run unit tests (fail)
4. Implement `processFindForm`
5. Run unit tests (pass)
6. Right-click on `processFindForm` > Write tests
7. Review created tests
8. Fix unintended behavior in created tests

```
public class OwnerControllerTes

…

@Test
public void testProcessFindForm_notFound() throws Exception {

  when(this.ownerRepository.findByLastName(or(isA(String.class), isNull())))
     .thenReturn(new ArrayList<Owner>());

  this.mockMvc.perform(get("/owners"))
     .andExpect(status().isOk());
     .andExpect(view().name("owners/findOwners"));
}
```

diffblue
AI for Code

TDD

1. git checkout a-test-tdd
2. Implement unit tests for `OwnerController.processFindForm`
3. Run unit tests (fail)
4. Implement `processFindForm`
5. Run unit tests (pass)
6. Right-click on `processFindForm` > Write tests
7. Review created tests
8. Fix unintended behavior in created tests
9. Amend implementation of `processFindForm`

```
public class OwnerController {
…

@GetMapping("/owners")
public String processFindForm(
    Owner owner, BindingResult result, Map<String, Object> model) {

  Collection<Owner> results = this.owners.findByLastName(owner.getLastName());
  if (results.isEmpty()) {
    result.rejectValue("lastName", "notFound", "not found");
    return "owners/findOwners";
  } else if (results.size() == 1) {
     owner = results.iterator().next();
     return "redirect:/owners/" + owner.getId();
  } else {
     model.put("selections", results);
     return "owners/ownersList";
  }
}
```

diffblue
AI for Code

TDD

1. git checkout a-test-tdd
2. Implement unit tests for `OwnerController.processFindForm`
3. Run unit tests (fail)
4. Implement `processFindForm`
5. Run unit tests (pass)
6. Right-click on `processFindForm` > Write tests
7. Review created tests
8. Fix unintended behavior in created tests
9. Amend implementation of `processFindForm`

```
public class OwnerControllerTes

…

@Test
public void testProcessFindForm1 throws Exception {
  Owner owner = new Owner();
  owner.setLastName("Doe");
  owner.setId(1);
  owner.setCity("Oxford");
  owner.setAddress("42 Main St");
  owner.setFirstName("Jane");
  owner.setTelephone("4105551212");
  when(this.ownerRepository.findByLastName(or(isA(Stri
      .thenReturn(Collections.singletonList(owner));

  this.mockMvc.perform(get("/owners"))
      .andExpect(status().is3xxRedirection())
      .andExpect(view().name("redirect:/owners/1"));
}
```

What's the intended behavior without "lastName" parameter?

diffblue
AI for Code

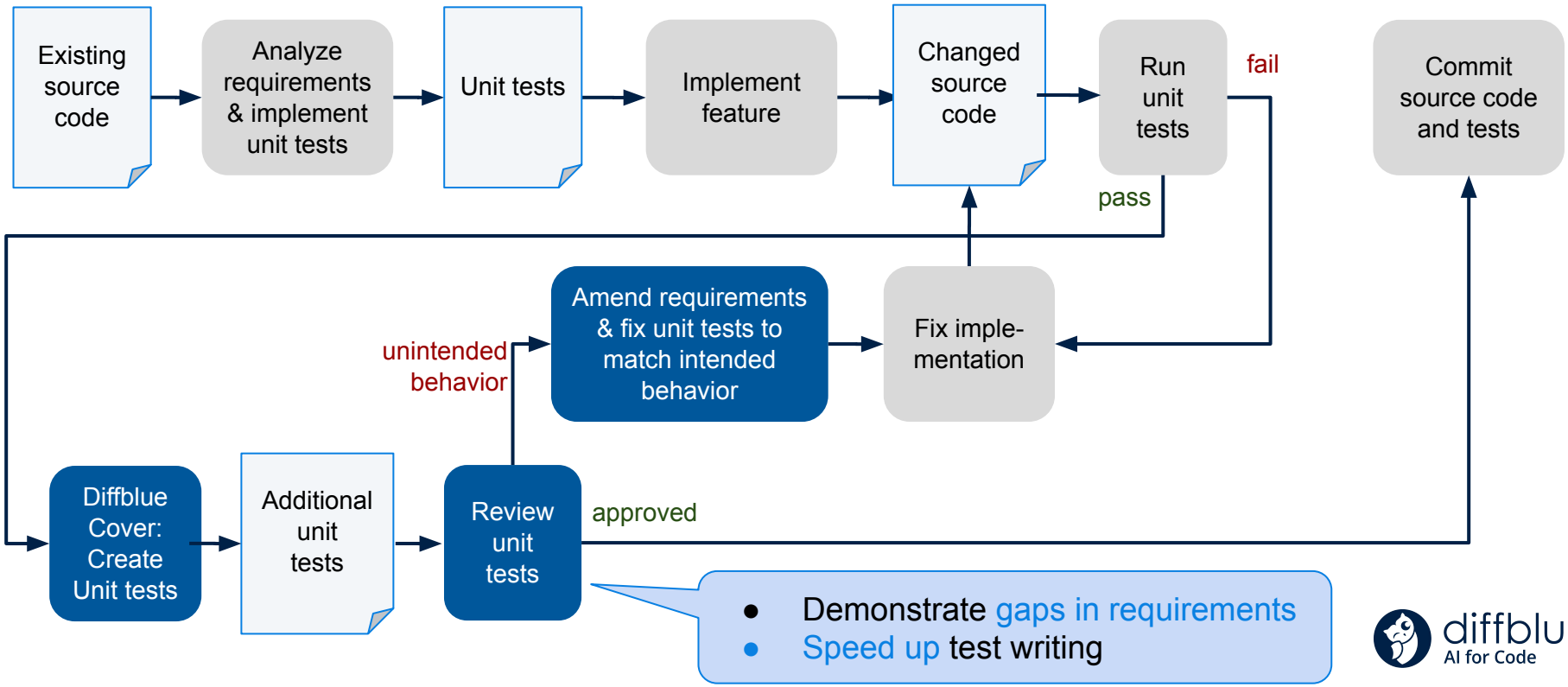
TDD

1. git checkout a-test-tdd
2. Implement unit tests for `OwnerController.processFindForm`
3. Run unit tests (fail)
4. Implement `processFindForm`
5. Run unit tests (pass)
6. Right-click on `processFindForm` > Write tests
7. Review created tests
8. Fix unintended behavior in created tests
9. Amend implementation of `processFindForm`

```
public class OwnerController {
…

@GetMapping("/owners")
public String processFindForm(
    Owner owner, BindingResult result, Map<String, Object> model) {
  // allow parameterless GET request for /owners to return all records
  if (owner.getLastName() == null) {
    owner.setLastName(""); // empty string signifies broadest possible search
  }
  Collection<Owner> results = this.owners.findByLastName(owner.getLastName());
  if (results.isEmpty()) {
    result.rejectValue("lastName", "notFound", "not found");
    return "owners/findOwners";
  } else if (results.size() == 1) {
     owner = results.iterator().next();
     return "redirect:/owners/" + owner.getId();
  } else {
     model.put("selections", results);
     return "owners/ownersList";
  }
```

diffblue
AI for Code

Test-Driven Development



diffblue
AI for Code

What’s more?

Diffblue Cover IntelliJ Plugin

- Community Edition: for open source only
- Paid plugin: for commercial software

Diffblue Cover CLI

- For integration into CI
- Try on https://www.diffblue.com/try-cover-browser

Takeaways

- **Diffblue Cover** is a tool that creates unit tests.
- Does not replace software developers
- Complements and speeds up manual test writing
- Created tests tell you what the current behavior
- Use cases:
  - **Refactoring**: demonstrate unintended changes
  - **New feature**: review demonstrates buggy implementation
  - **TDD**: review demonstrates gaps in requirements

Giving Feedback

shorturl.at/auHLM

Please fill me in!

Getting Support

https://forum.diffblue.com

I'd like to have this feature.

Cool stuff!

I don't get tests!

diffblue
AI for Code