LUXOFT TRAINING

Разработка через тестирование

Test Driven Development

Ivan Dyachenko <IDyachenko@luxoft.com>

Module 2

Для кого этот тренинг?

LUXOFT Training Center

1. **Beginner**
   Хорошая точка входа
2. **Intermediate**
   Поможет лучше всё структурировать в голове и объяснять коллегам
3. **Advanced**
   Можно использовать для обучения и проверки других

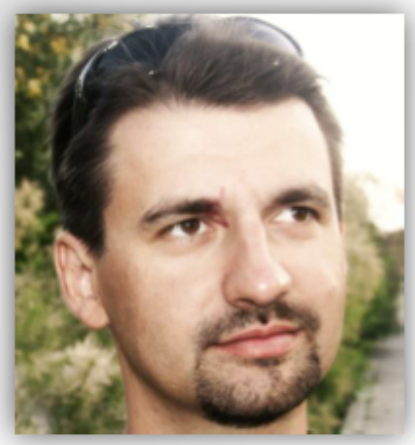
Содержание

LUXOFT Training Center

LUXOFT Training Center

Тесты на поведение супротив тестов на состояние

Пример

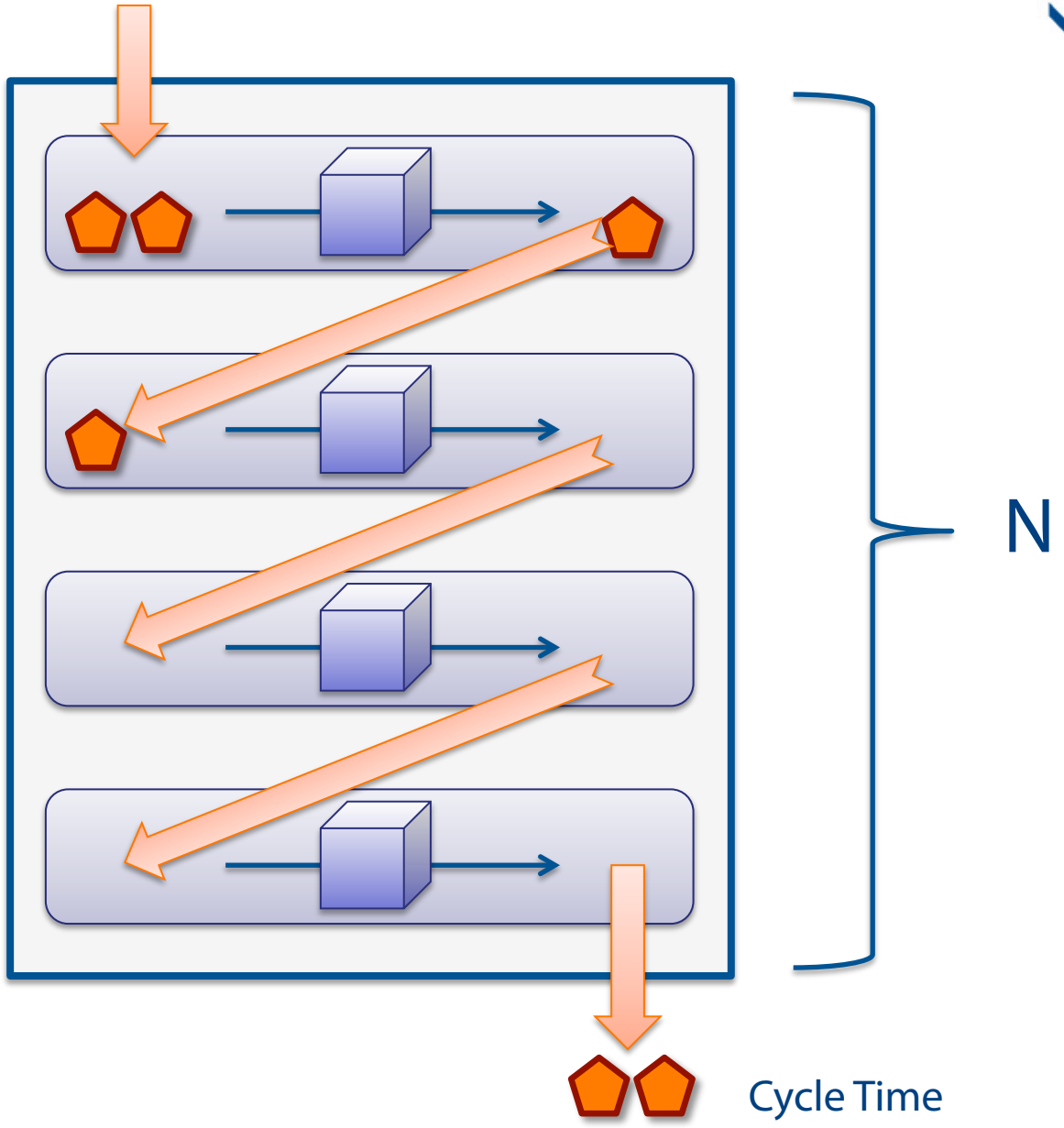


Андрей Бибичев

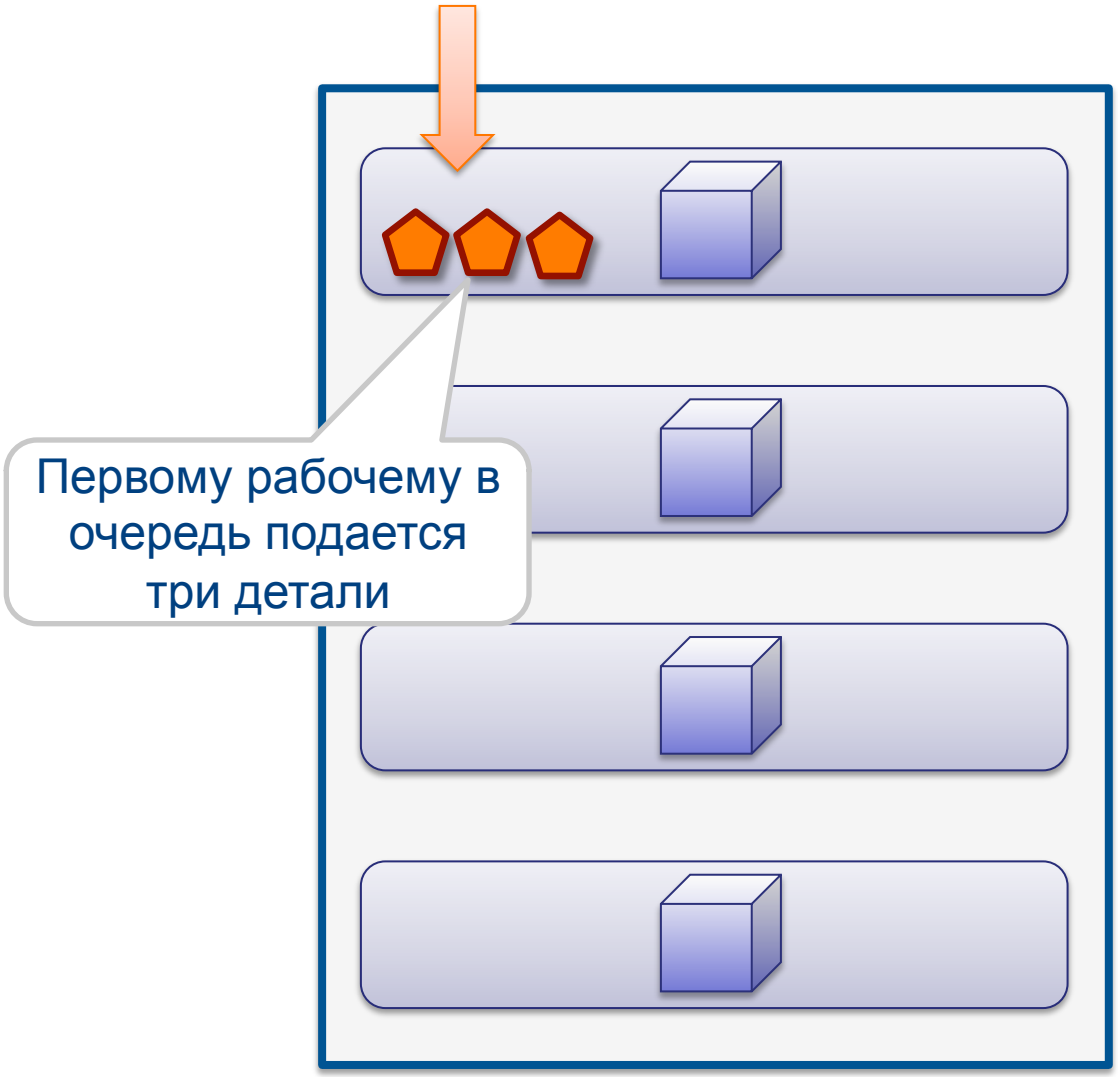
Conveyor - сравнение тестов на поведение и тестов на состояние

http://www.slideshare.net/bibigine

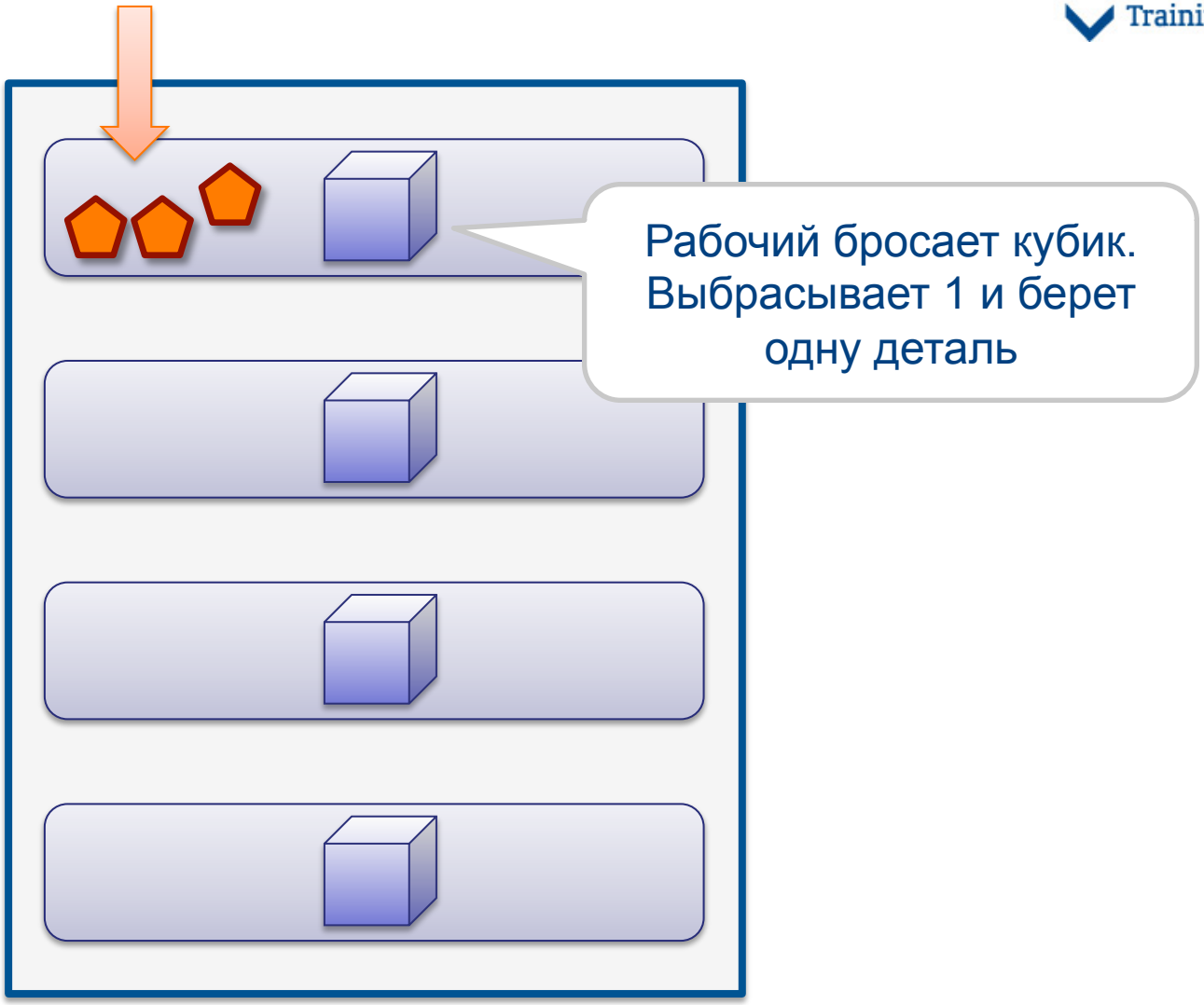
Конвейер



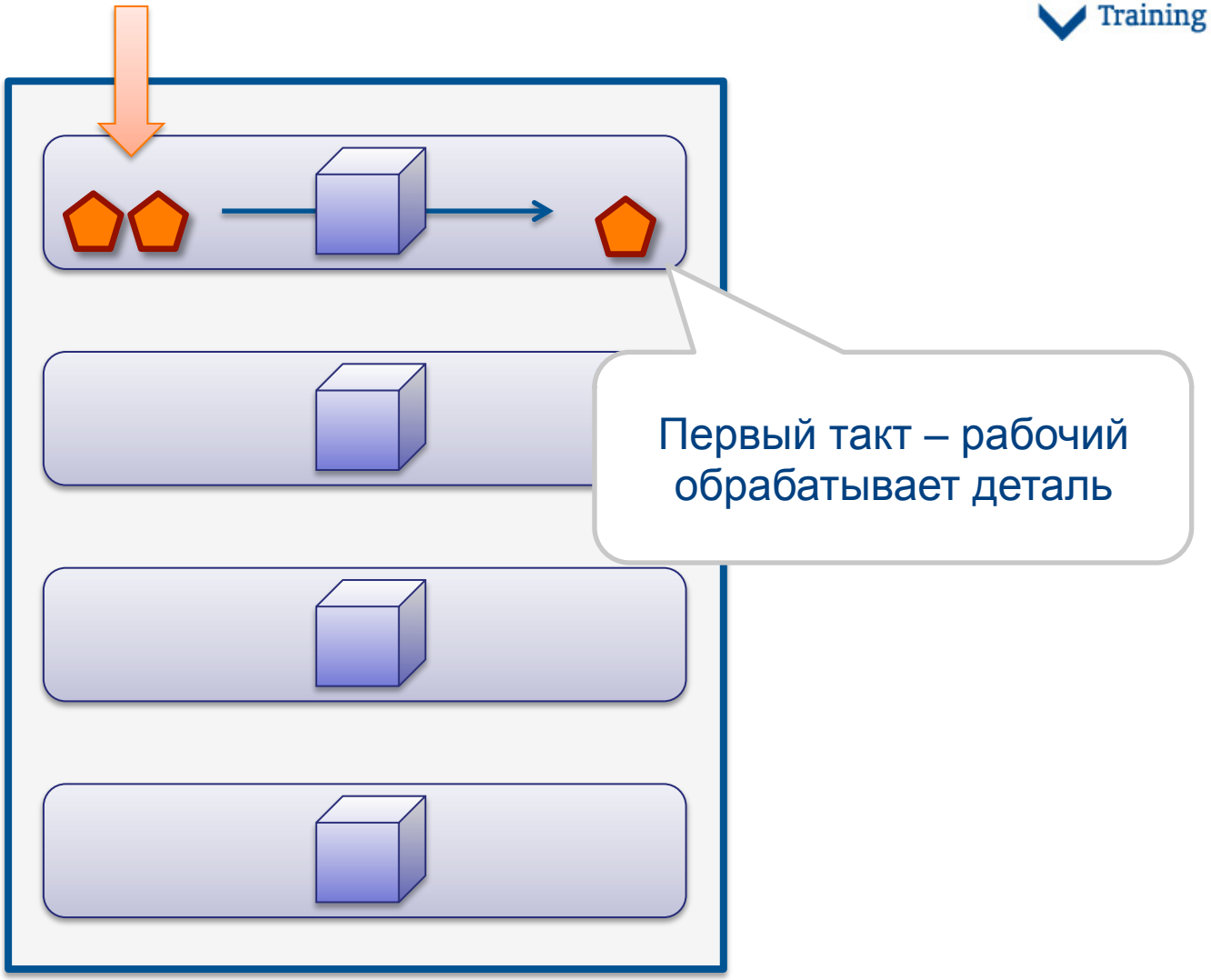
Контрольный пример



Контрольный пример

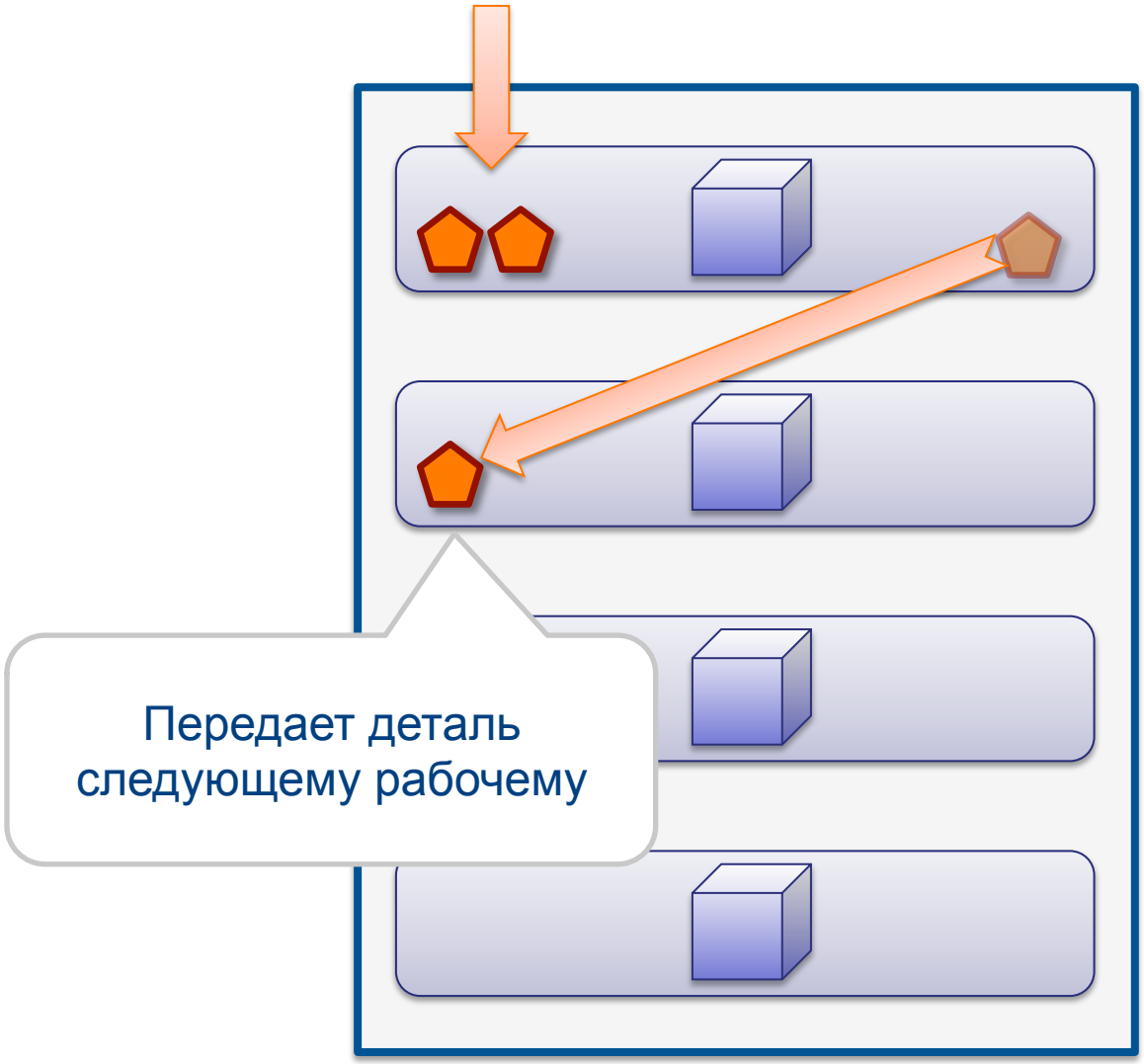


Контрольный пример

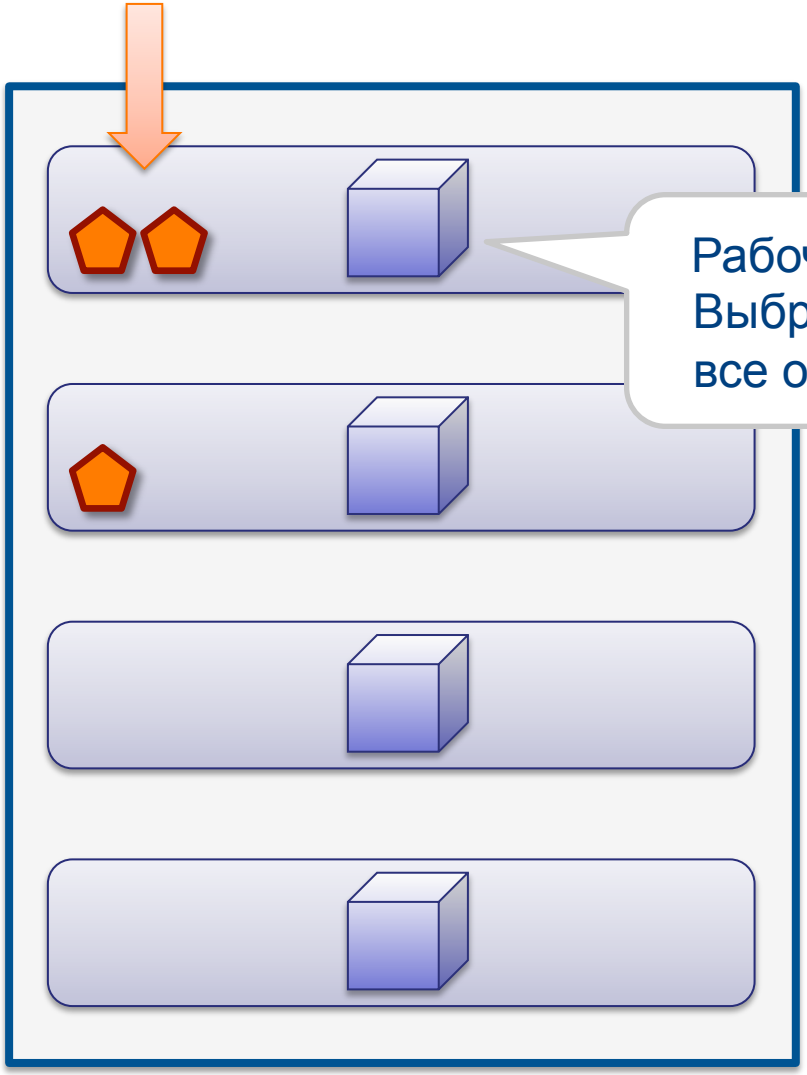


Контрольный пример

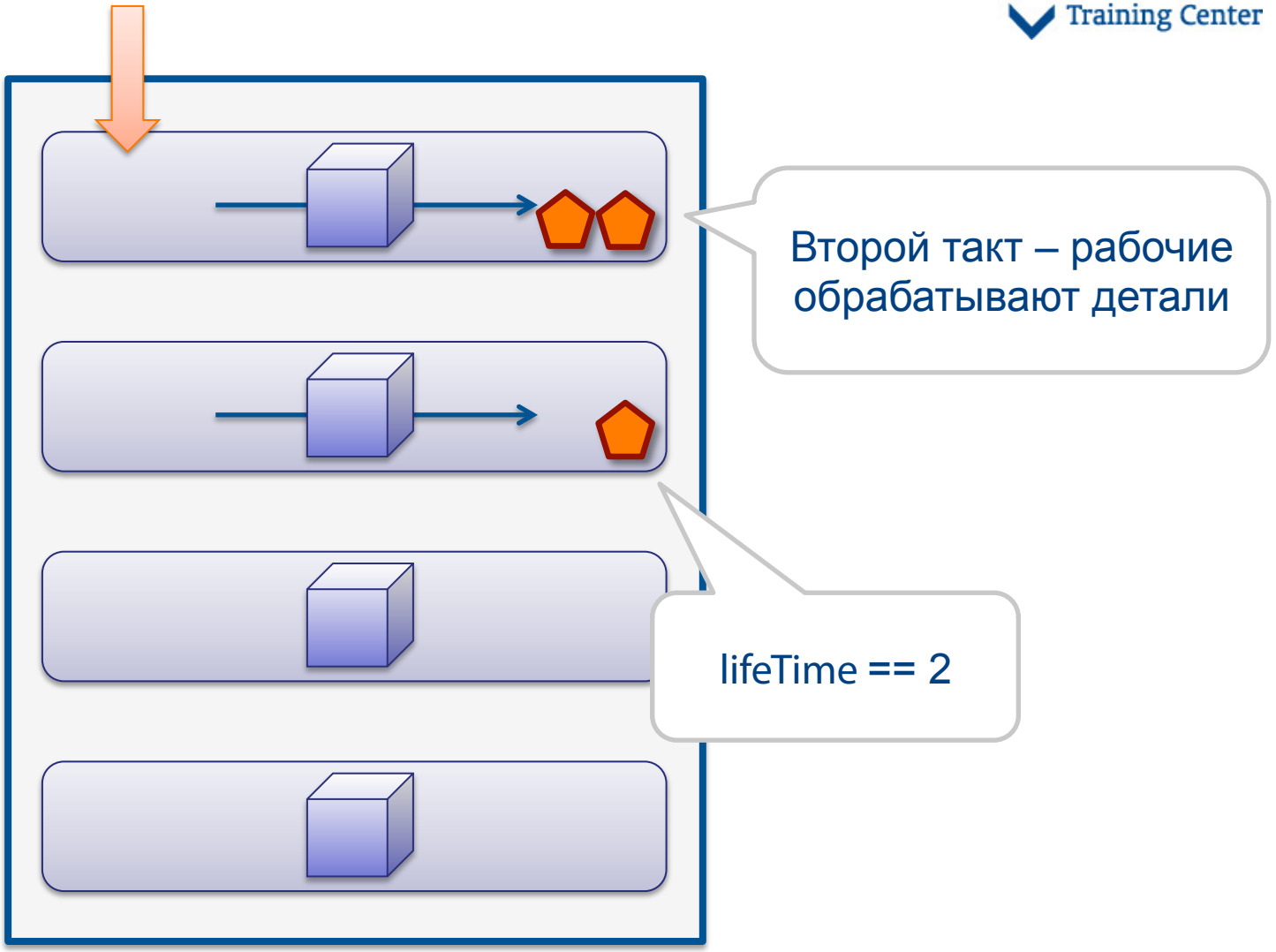
LUXOFT Training Center



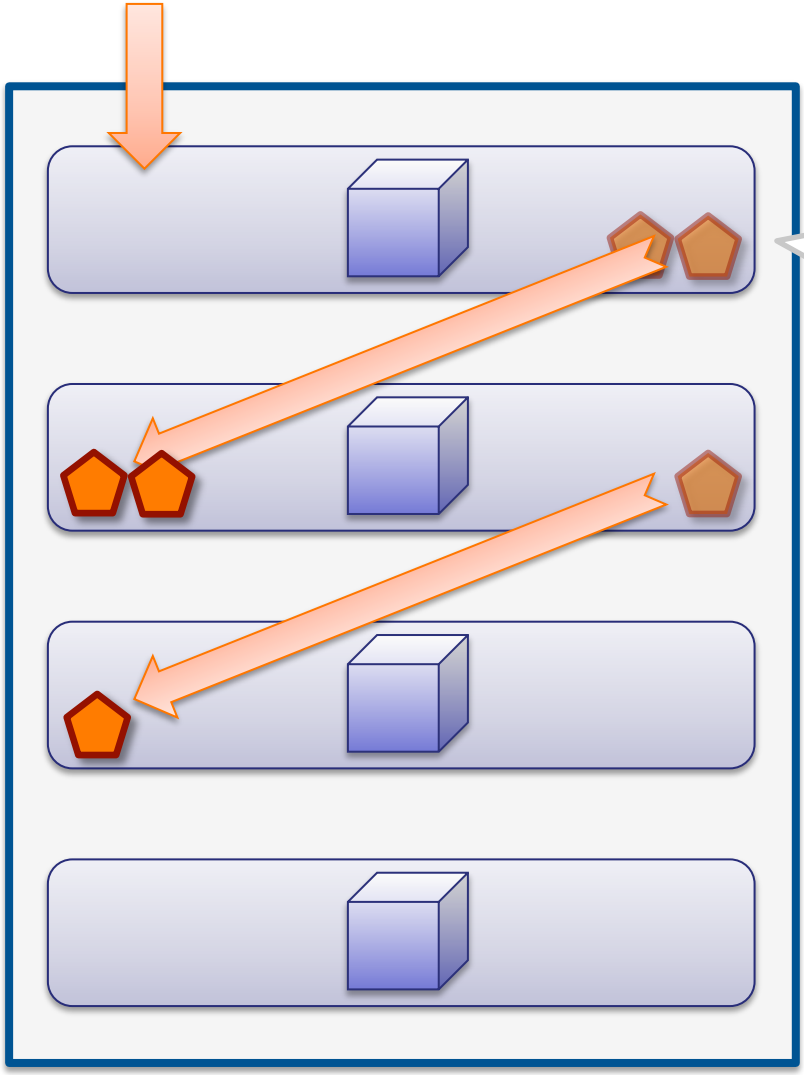
Конвейер



Конвейер

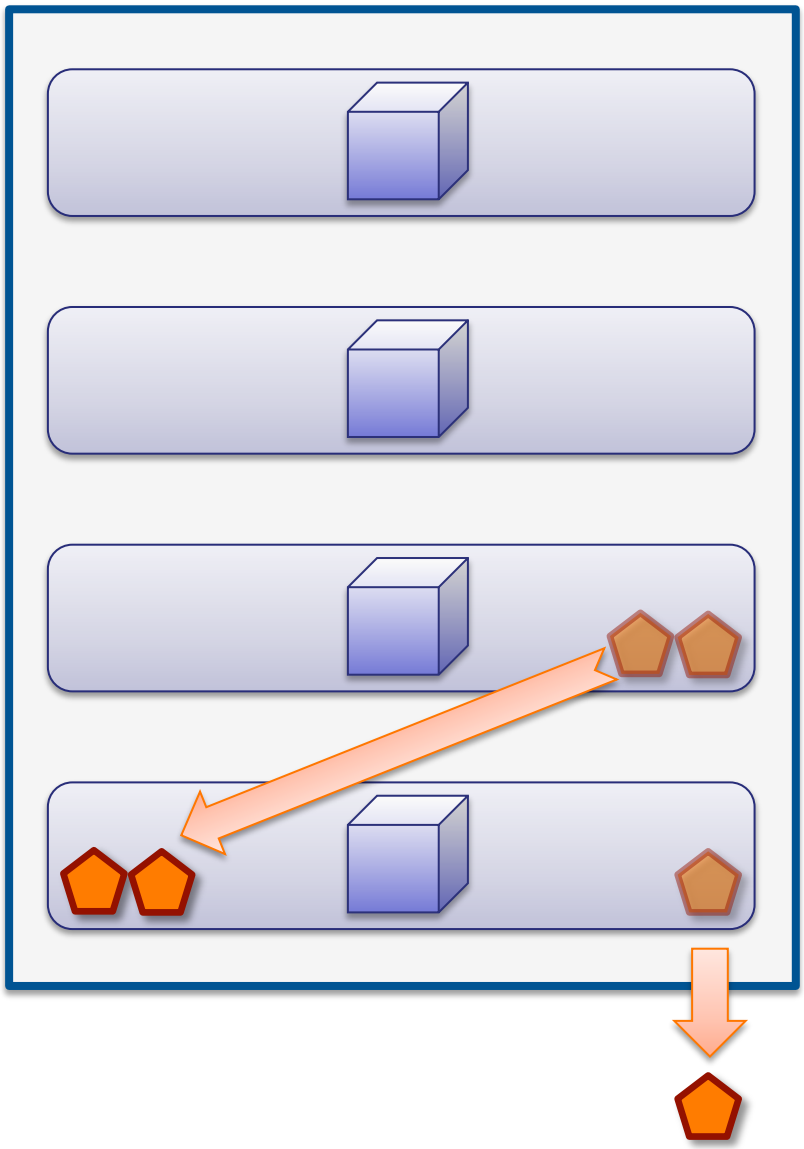


Конвейер



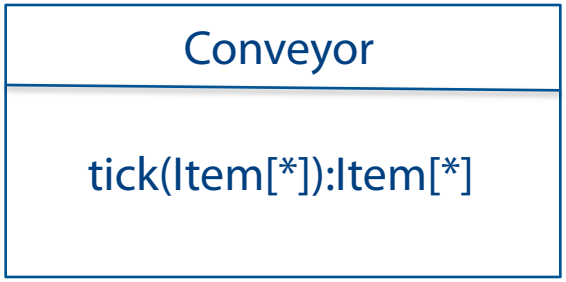
Передают детали дальше

Конвейер



То, что обработал последний рабочий, является выходом конвейера за соответствующий цикл

Надо написать

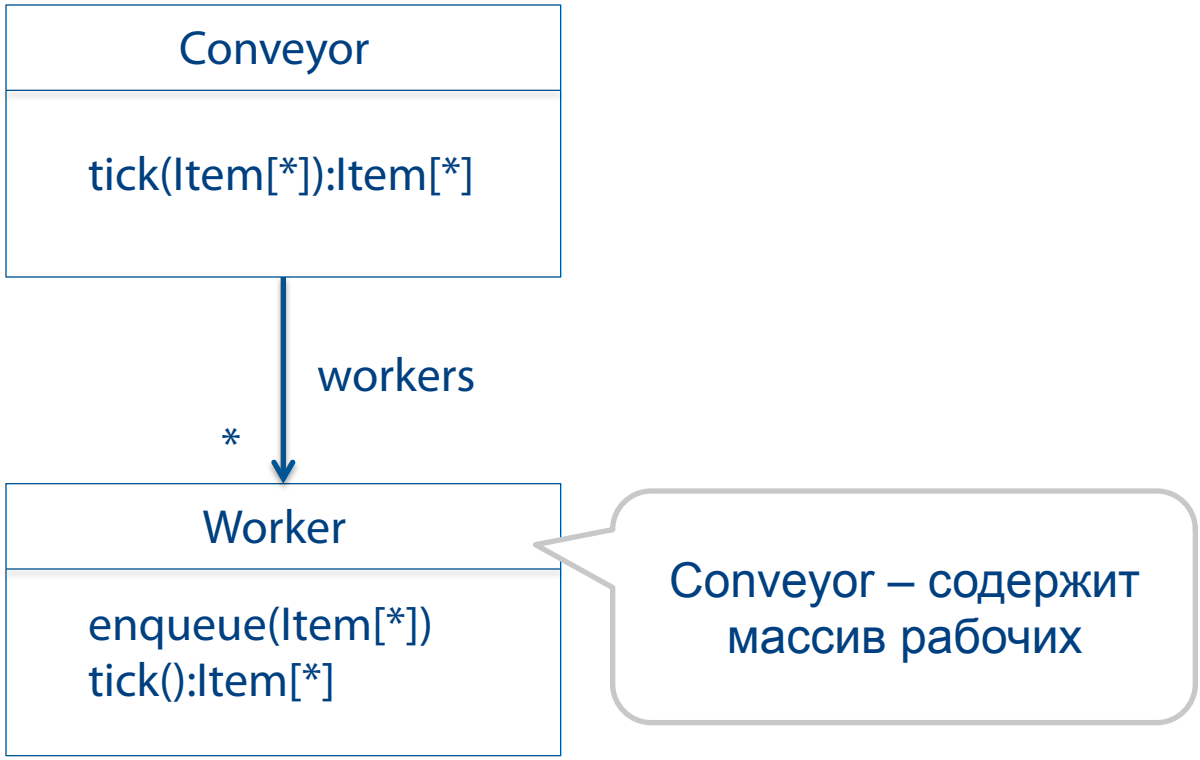


Написать класс Conveyor, который имеет один метод

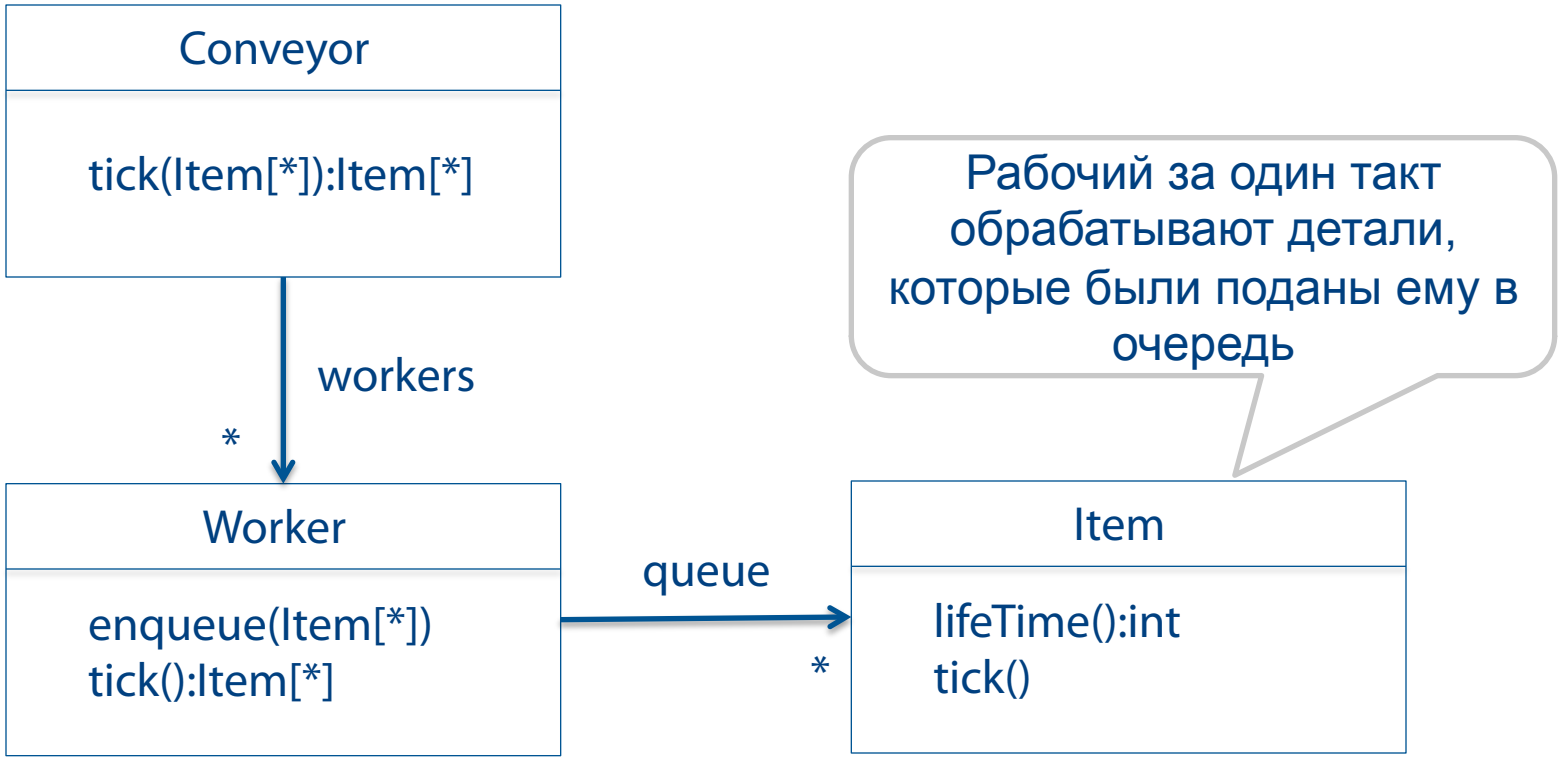
- tick(Item[*]):Item[*] – вызывается каждый такт. Параметром передается детали на входную очередь первого рабочего.
- Возвращает массив обработанных деталей - выход конвейера за соответствующий цикл

A quick design session

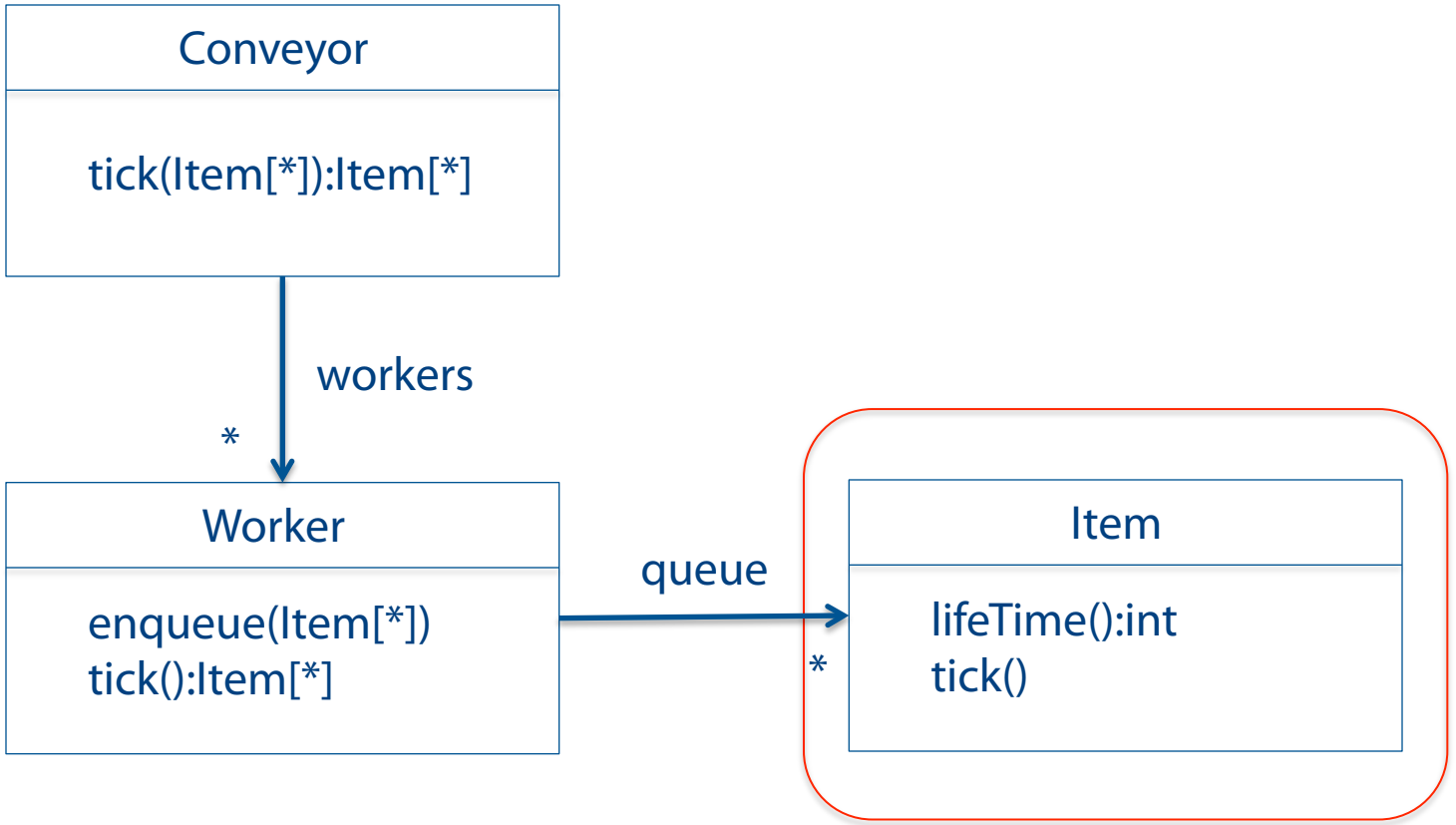
A quick design session



A quick design session



A quick design session



Item

LUXOFT Training Center

У вновь созданной детали

время жизни должно равняться нулю

- дано: новая деталь
- когда: запрашиваем у нее время жизни
- тогда: получаем в результате 0

Пишем тест

LUXOFT Training Center

```
package conveyor;

import org.junit.Test;
import static org.fest.assertions.Assertions.assertThat;

public class ItemTest {
    @Test
    public void shouldHaveZeroLifeTimeAfterCreation() {
        // given
        final Item item = new Item();
        // when
        int lifeTime = item.lifeTime();
        // then
        assertThat(lifeTime).isZero();
    }
}
```

FEST

Fixtures for Easy Software Testing


```
int removed = employees.removeFired();
assertThat(removed).isZero();

List<Employee> newEmployees = employees.hired(TODAY);
assertThat(newEmployees).hasSize(6)
                        .contains(frodo, sam);

assertThat(yoda).isInstanceOf(Jedi.class)
                .isEqualTo(foundJedi)
                .isNotEqualTo(foundSith);
```

FEST

LUXOFT Training Center

Это «вырожденный» тест на состояние

Пишем минимум кода

LUXOFT Training Center

```
public class Item {
    public int lifeTime() {
        return 0;
    }
}
```

Дано:

арбуз + гиря 1 кг = гиря 6 кг
арбуз = ?

Решение:

x + 1 = 6
x = 6 – 1 = 5

Ответ: 5 кг

Шаблон теста

```
Test...
{
    // Arrange
    ...
    // Action
    ...
    // Assetion
    ...
}
```

```
Should...
{
    // Given
    ...
    // When
    ...
    // Then
    ...
}
```

Критерий хорошо оформленного теста

LUXOFT Training Center

- Содержательное название
- Короткое тело (max = 20-30 строк)
- По шаблону AAA или GIVEN-WHEN-THEN
- Без циклов
- Без ветвлении (if-ов и case-ов)
- Должен легко читаться (literate programming)

Следующий тест

Оповещение о том, что прошел такт конвейера должно увеличивать значение времени жизни на один

- дано: новая деталь
- когда: оповещаем ее о такте конвейера
- тогда: время жизни становится 1

Пишем тест

LUXOFT Training Center

```
@Test
public void shouldIncrementLifeTimeDuringTick() {
    // given
    final Item item = new Item();
    // when
    item.tick();
    // then
    assertThat(item.lifeTime()).isEqualTo(1);
}
```

LUXOFT Training Center

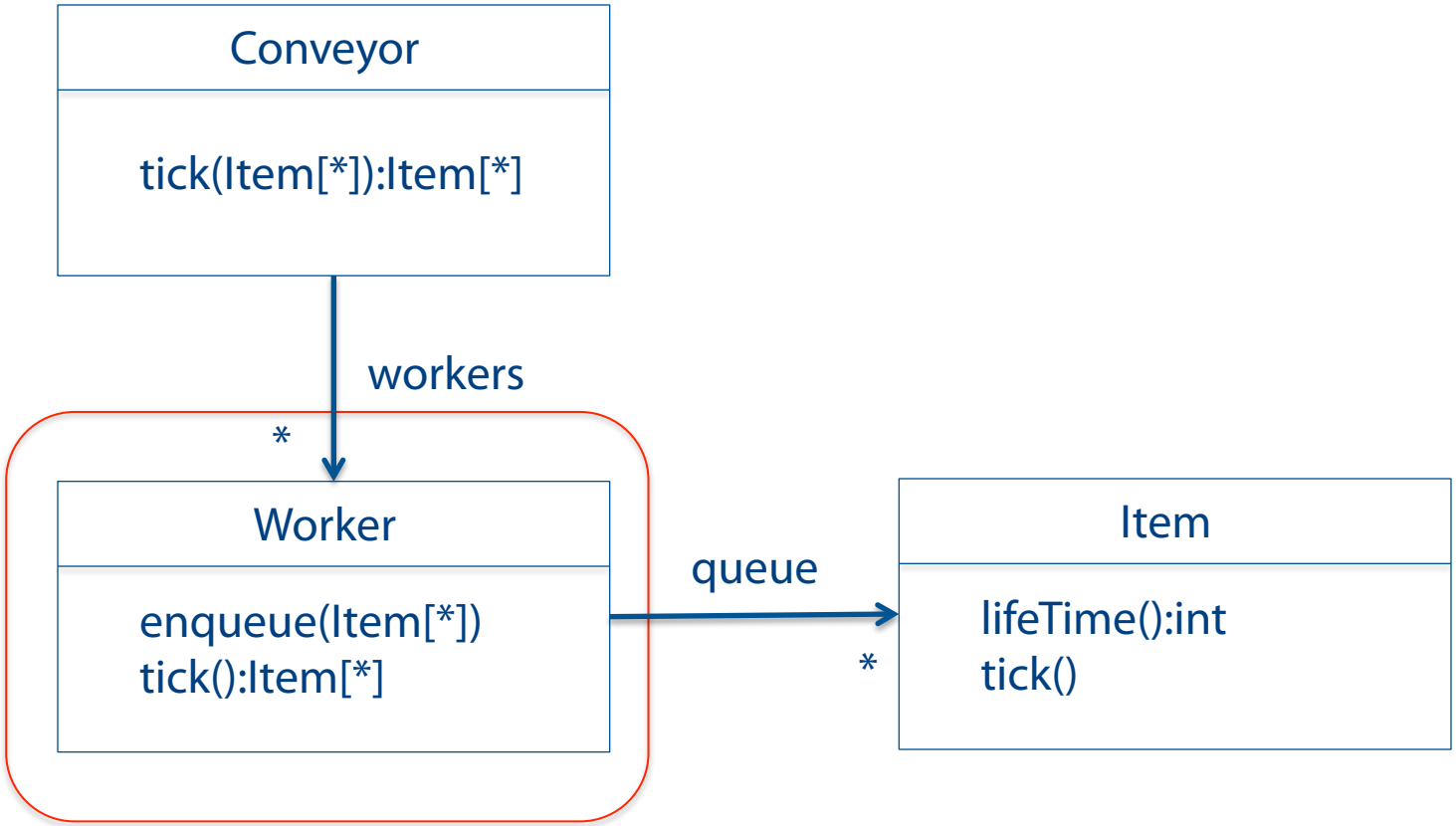
Это примитивный пример
теста на состояние

Пишем код

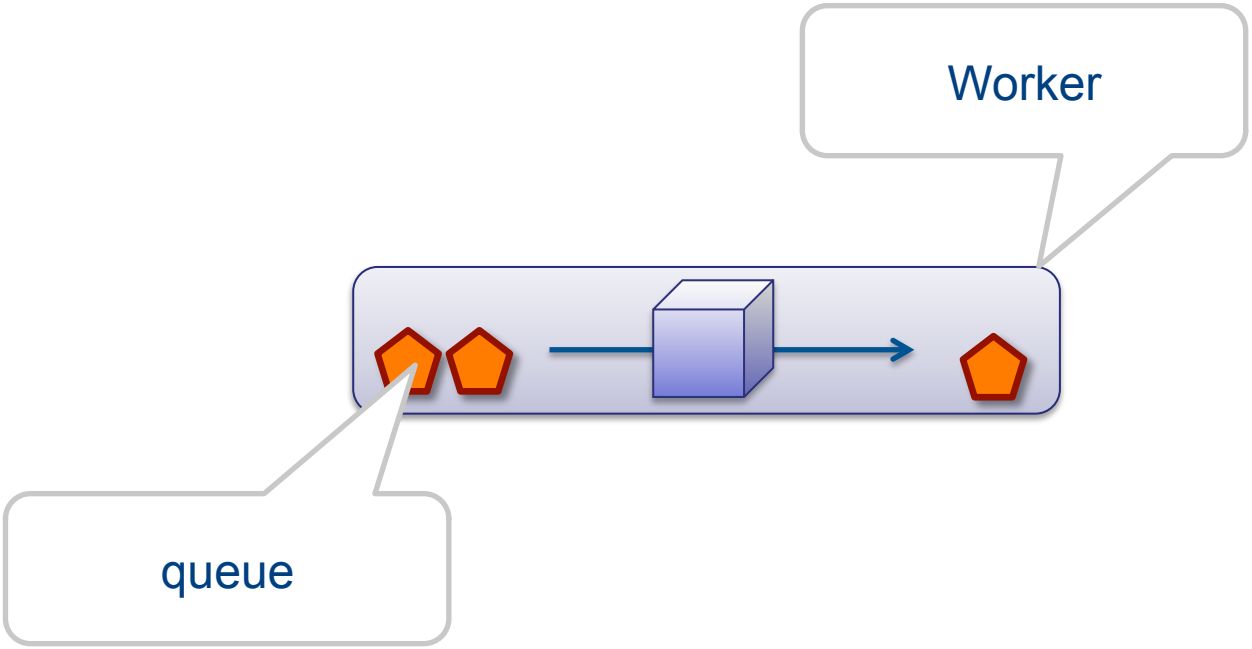
LUXOFT Training Center

```
public class Item {
    private int lifeTime;

    public int lifeTime() {
        return lifeTime;
    }

    public void tick() {
        lifeTime++;
    }
}
```

Переходим к Worker



Worker



Worker

LUXOFT Training Center

- Рабочий ничего не обрабатывает, если нет деталей
- У вновь созданного рабочего входная очередь деталеий пуста

Пишем тест

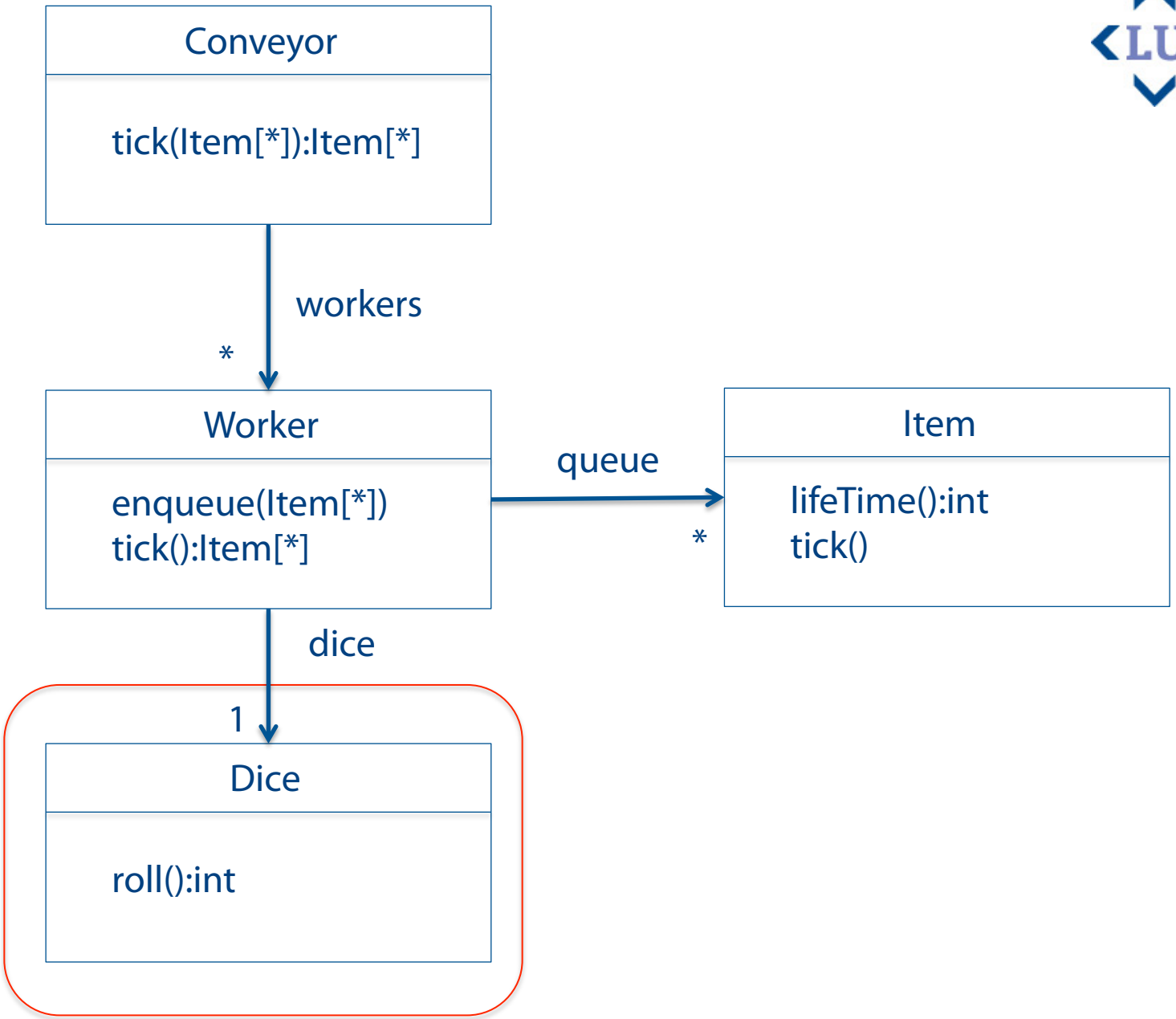
LUXOFT Training Center

```
public class WorkerTest {
    @Test
    public void shouldReturnNothingIfNothingToDo() {
        // given
        final Worker worker = new Worker();
        // when
        final List<Item> output = worker.tick();
        // then
        assertThat(output).isEmpty();
    }
}
```

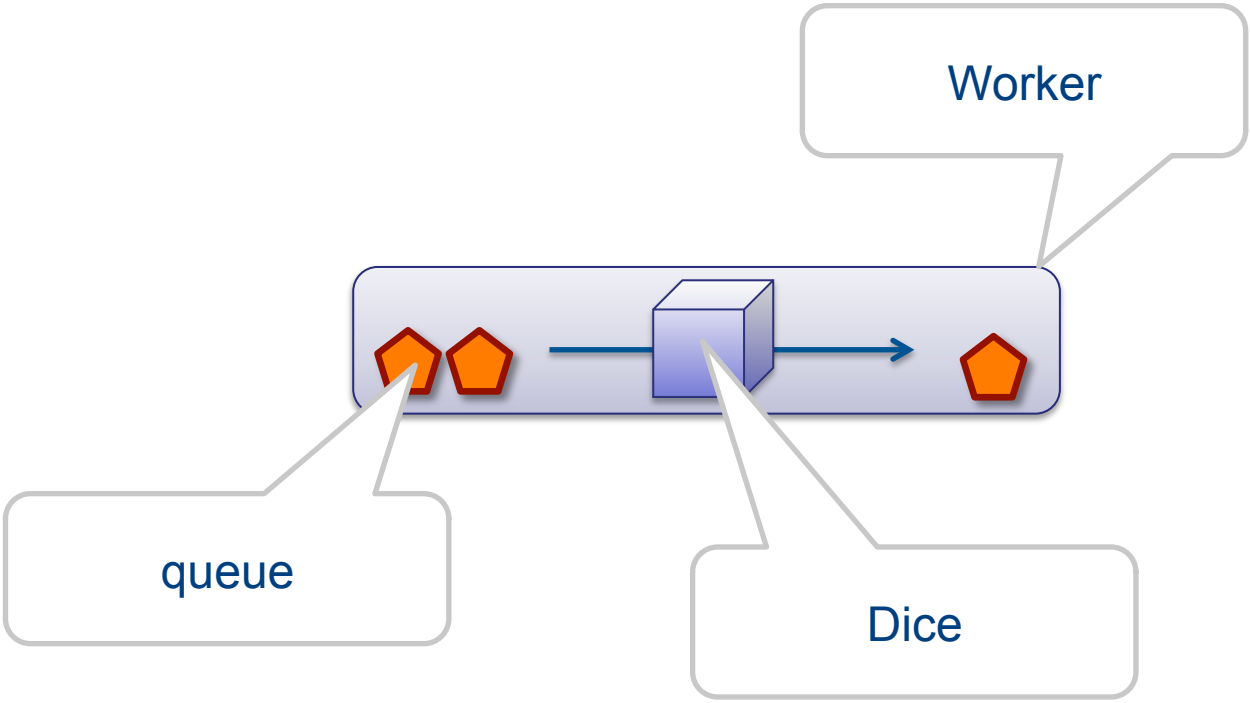
Worker

Если во время обработки на кубике выпало значение большее количества деталеий в очереди,

то рабочиий обрабатывает все детали в очереди (и больше ничего)



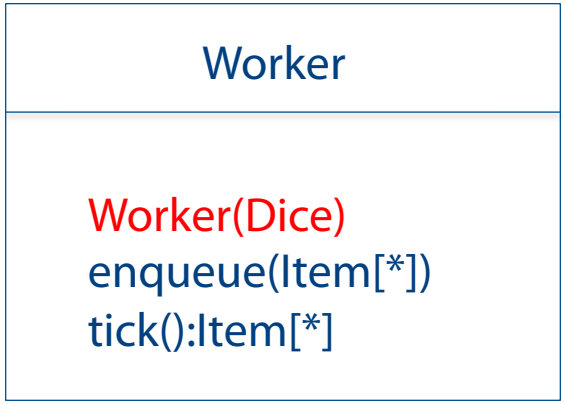
Worker



Dependency Injection (DI)

LUXOFT Training Center

Dependency Injection (DI) через конструктор

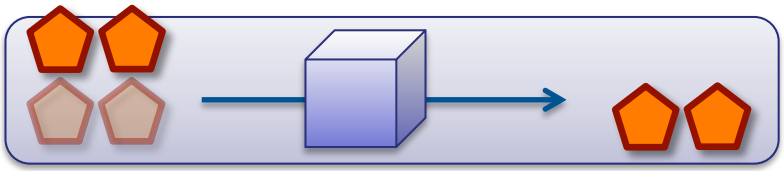


Пишем тест

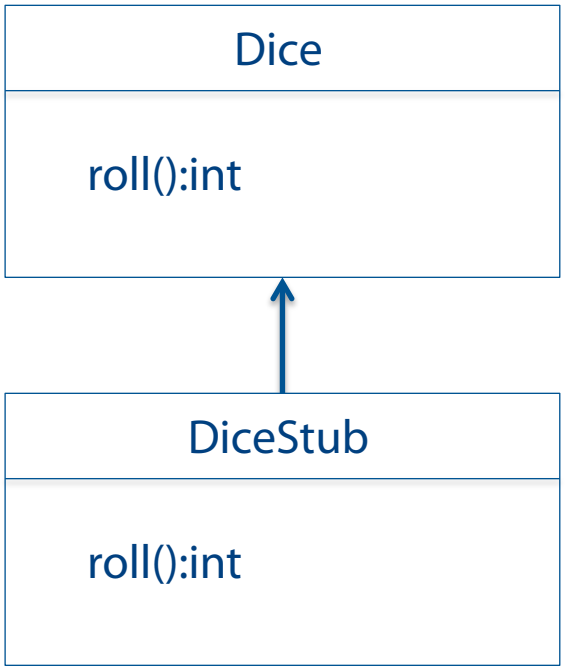
LUXOFT Training Center

```
@Test
public void shouldProcessNotGreaterThanItemsInQueue() {
    // given
    final Dice dice = createDiceStub(4);
    final Worker worker = new Worker(dice);
    final List<Item> items = Arrays.asList(
                                    new Item(),
                                    new Item());
    worker.enqueue(items);
    // when
    final List<Item> output = worker.tick();
    // then
    assertThat(output).isEqualTo(items);
}
```

Stub



enqueue tick



Mock

```
private Dice createDiceStub(int rollValue) {
    final Dice dice = mock(Dice.class);
    when(dice.roll()).thenReturn(rollValue);
    return dice;
}
```

LUXOFT Training Center

moq
test first. mock me later.

Хотя мы и воспользовались mock-объектом,
это всё равно, по большому счету, тест на состояние

Worker

Если во время обработки на кубике выпало значение N, меньше количества деталей в очереди, то обрабатывается только первые N деталей из очереди

Worker

LUXOFT Training Center

```
@Test
public void shouldProcessNotGreaterThanRolledValue() {
    // given
    final int rollValue = 3;
    final Dice dice = createDiceStub(rollValue);
    final Worker worker = new Worker(dice);
    final List<Item> items = Arrays.asList(
                    new Item(), new Item(), new Item(), new Item());
    worker.enqueue(items);
    // when
    final List<Item> output = worker.tick();
    // then
    assertThat(output).isEqualTo(items.subList(0, rollValue));
}
```

Еще аналогичные тесты:

LUXOFT Training Center

- Проверяем, что enqueue() **добавляет** в очередь
- Проверяем, что tick() **удаляет** из очереди обработанные детали

Тупой, но важный тест:

Во время Worker.tick() кубик бросается ровно один раз!

Кубик бросается один раз

LUXOFT Training Center

```
@Test
public void shouldRollDiceOnlyOnceDuringTick() {
    // given
    final Dice dice = createDiceStub(3);
    final Worker worker = new Worker(dice);
    final List<Item> items = Arrays.asList(
                                    new Item(), new Item(),
                                    new Item(), new Item());
    worker.enqueue(items);
    // when
    worker.tick();
    // then
    verify(dice, times(1)).roll();
}
```

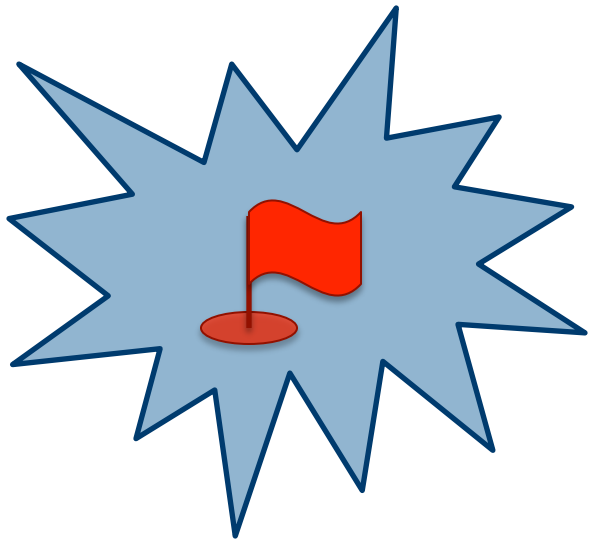
Тест на поведение

Это примитивный пример **теста на поведение**:
мы проверили как взаимодействует наш объект с другим объектом.

Тесты

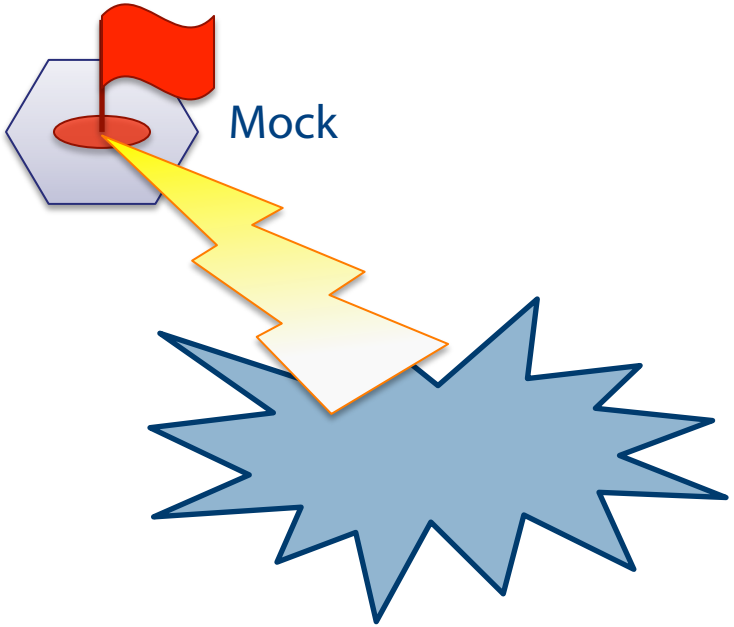
LUXOFT Training Center

На состояние



Object

На поведение



Object

Закрепим материал:

LUXOFT Training Center

- Проверим, что во время Worker.tick() вызывается Item.tick() для всех деталей, находящихся в очереди на начало tick()-а
- Это можно проверить, не прибегая к mock-ам – через значение lifeTime(), но тогда мы тестируем два класса сразу, а не один в изоляции

Закрепим материал:

LUXOFT Training Center

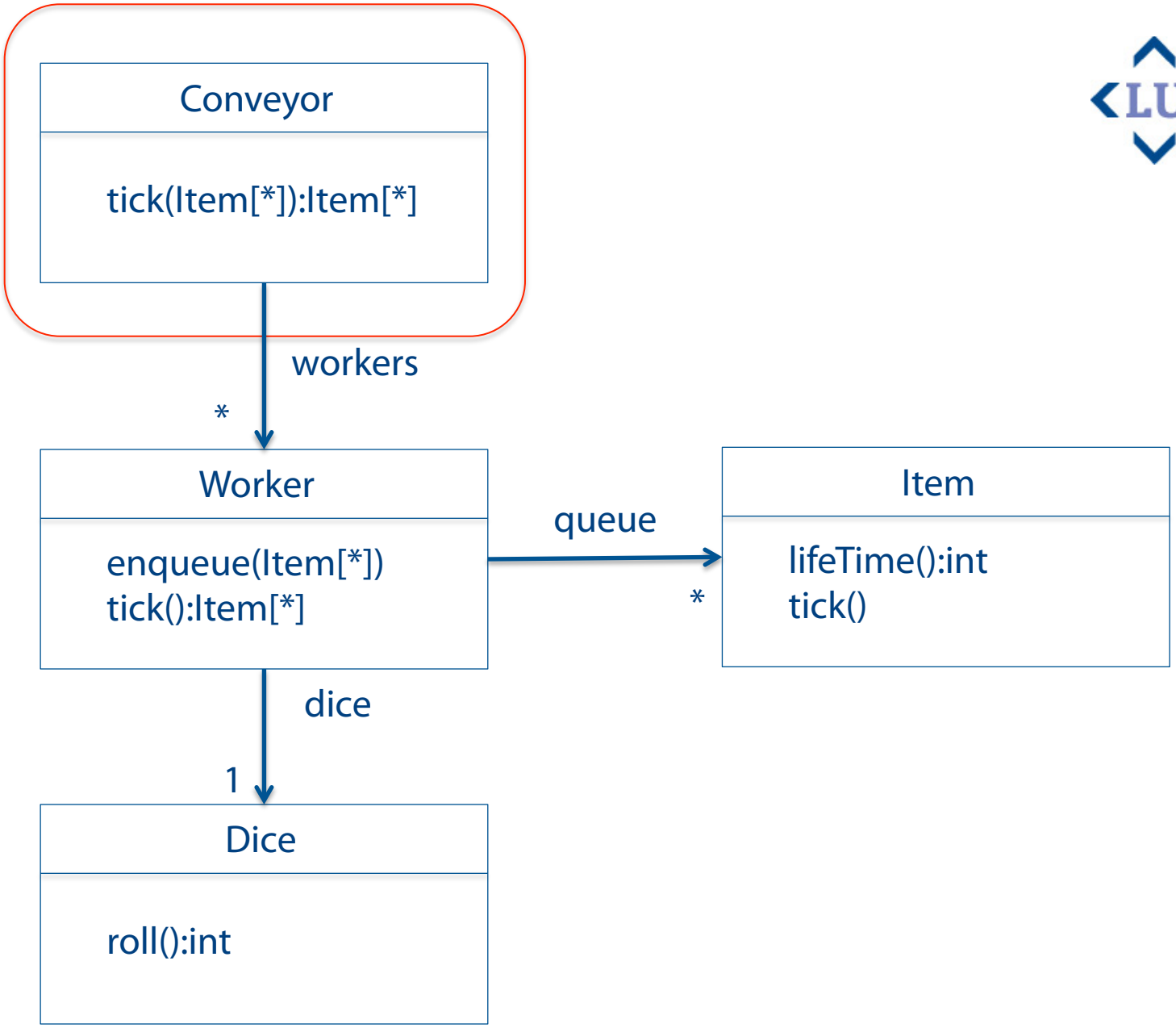
```
@Test
public void shouldCallTickForAllItemsInQueue() {
    // given
    final Dice dice = createDiceStub(2);
    final Worker worker = new Worker(dice);
    final Item firstItem = mock(Item.class);
    final Item secondItem = mock(Item.class);
    final List<Item> items = Arrays.asList(firstItem, secondItem);
    worker.enqueue(items);
    // when
    worker.tick();
    // then
    verify(firstItem, times(1)).tick();
    verify(secondItem, times(1)).tick();
}
```

Совет «по случаю»

- Избегайте имен переменных item1, item2 и т.п.
- Точно запутаетесь и опечатаетесь
- Лучше говорящие имена
- Или на худоий конец: firstItem, secondItem и т.п.

LUXOFT Training Center

Переходим к самому интересному



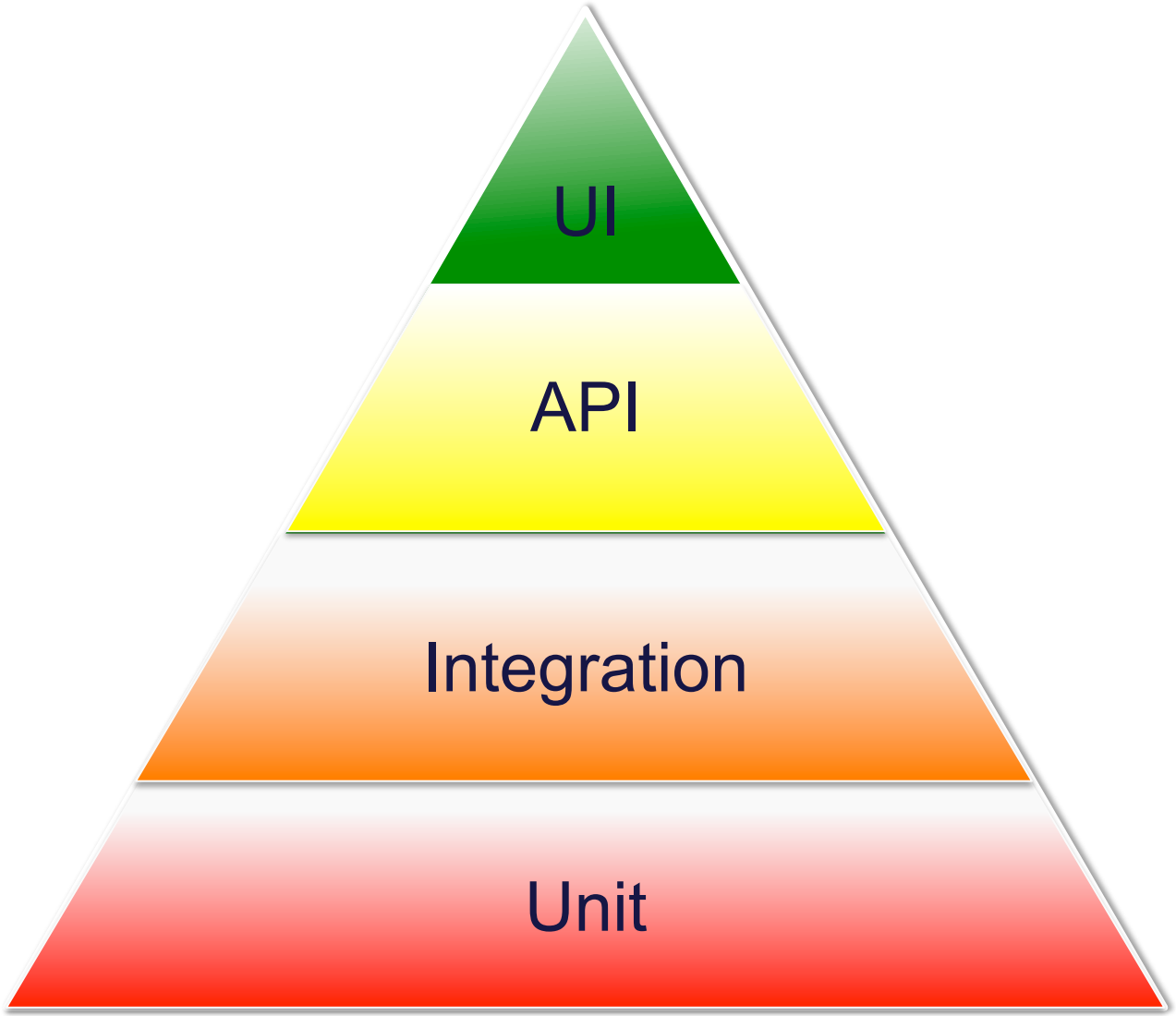
LUXOFT Training Center

Как тестировать?

Уровни качества

Пирамида автоматизации

LUXOFT Training Center



Вопросы ?

Разработка через тестирование

IDyachenko@luxoft.com