

# 計算機科学第一（講義）

## 単体テスト，テスト駆動開発

脇田建

---

2015.10.13

# 講義資料の入手方法

---

- ❖ 先週、作成した lecture ディレクトリ（build.sbt というファイルがあるところ）に移動してから以下のコマンドを実行

```
git pull
```



# 小テスト

---

# テスト駆動開発

---

- ❖ ひとまず、やる気のないコードを作成
- ❖ テストのためのコードを作成（完璧でなくてよい）
- ❖ 以下を繰り返す
  - ❖ テストを実行 → テストに失敗したら修復
  - ❖ バグを発見 → バグを再現するテストを追加



# 例：閏年の計算

---

- ❖ 目標：西暦(Y)が与えられたときに，その年が閏年か否かを答えるメソッド`leapYear`を作成しなさい。

# 日本における閏年の根拠法

## 明治三十一年勅令第九十号

---

- ❖ 明治三十一年勅令第九十号（閏年ニ関スル件・明治三十一年五月十一日勅令第九十号
- ❖ 神武天皇即位紀元年数ノ四ヲ以テ整除シ得ヘキ年ヲ閏年トス
- ❖ 但シ紀元年数ヨリ六百六十ヲ減シテ百ヲ以テ整除シ得ヘキモノノ中更ニ四ヲ以テ商ヲ整除シ得サル年ハ平年トス



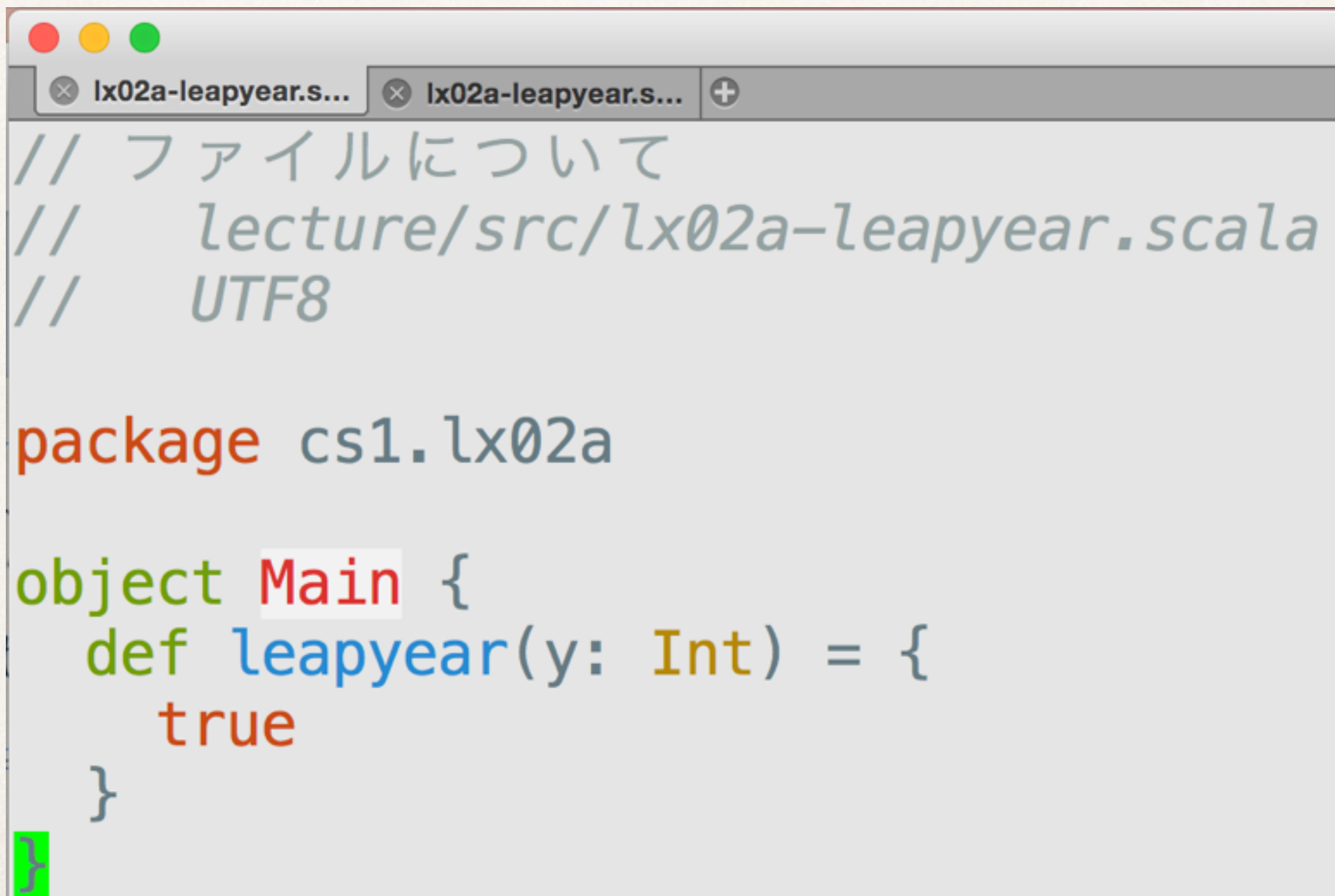
# 皇紀はつらいので 西暦で解釈すると

---

- ❖ グレゴリオ暦では、次の規則に従って400年間に（100回ではなく）97回の閏年を設ける。
- ❖ 西暦年が4で割り切れる年は閏年
- ❖ ただし、西暦年が100で割り切れる年は平年
- ❖ ただし、西暦年が400で割り切れる年は閏年

ステップ1：これ以上はないほど愚かなコードで始める。当然，バグを含む。型だけは仕様に合わせる。

---



```
// ファイルについて
//    lecture/src/lx02a-leapyear.scala
//    UTF8

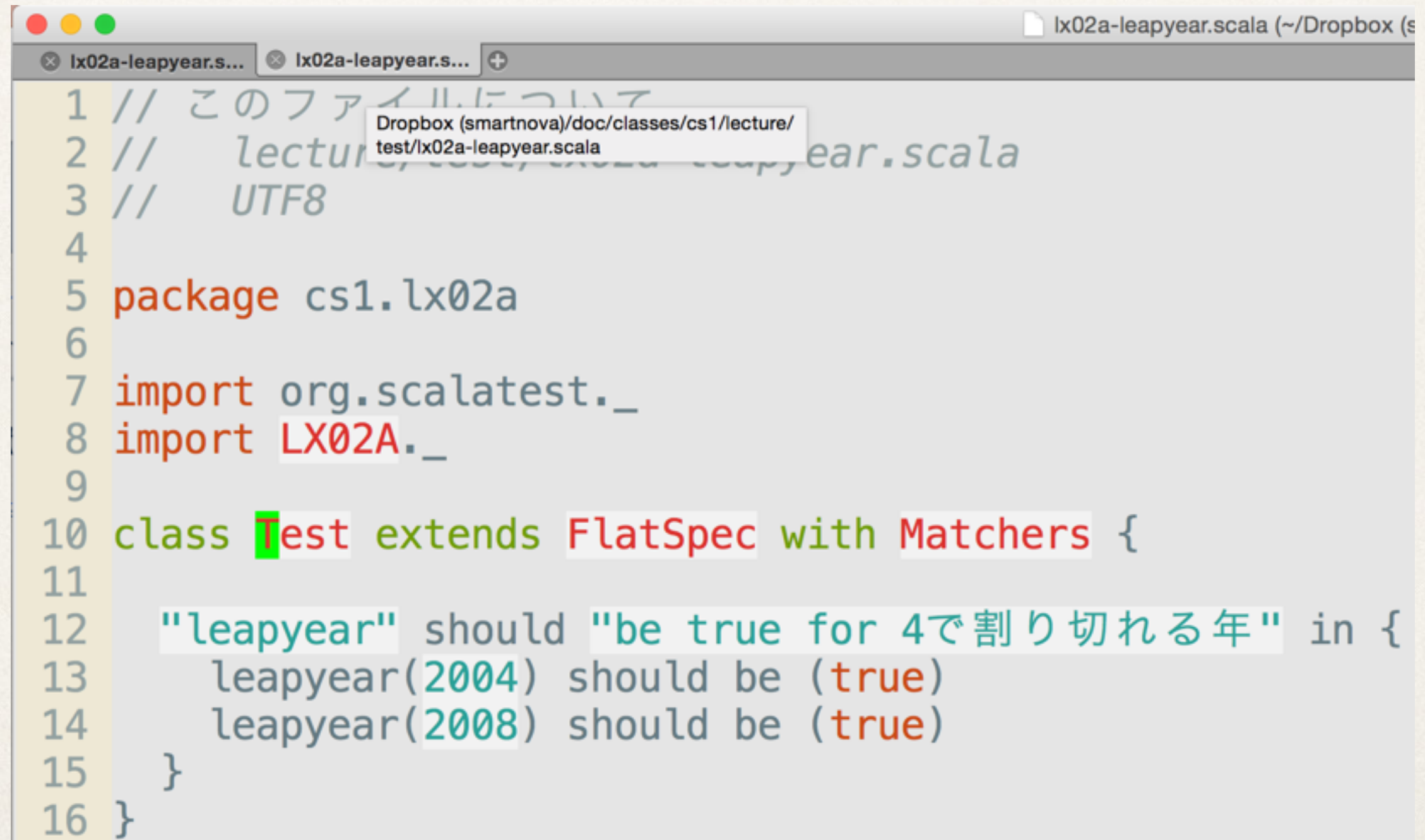
package cs1.lx02a

object Main {
  def leapyear(y: Int) = {
    true
  }
}
```



# ステップ2: テストのためのコードを作成 完璧でなくてよい

---



The screenshot shows a code editor window with the title bar "lx02a-leapyear.scala (~/Dropbox (s)". The editor contains the following Scala code:

```
1 // このファイルについて
2 //   lecture, test, lx02a-leapyear.scala
3 //   UTF8
4
5 package cs1.lx02a
6
7 import org.scalatest._
8 import LX02A._
9
10 class Test extends FlatSpec with Matchers {
11
12     "leapyear" should "be true for 4で割り切れる年" in {
13         leapyear(2004) should be (true)
14         leapyear(2008) should be (true)
15     }
16 }
```

A tooltip is visible over the code, displaying the file path: "Dropbox (smartnova)/doc/classes/cs1/lecture/test/lx02a-leapyear.scala".

# ステップ2: テストのためのコードを作成 完璧でなくてよい

```
1 // このファイルについて
2 //   lecture, test, scala
3 //   UTF8
4
5 package cs1.lx02a
6
7 import org.scalatest._
8 import Main._
9
10 class Test extends FlatSpec with Matchers {
11
12     "leapyear" should "be true for 4で割り切れる年" in {
13         leapyear(2004) should be (true)
14         leapyear(2008) should be (true)
15     }
16 }
```

テスト対象の object の名前  
“import Main.\_” 宣言により,  
Main.leapyear でなく 単に  
leapyear と参照できる



# sbtコマンドを起動してテストを実行

## Scala Build Tool

---

```
1. Default
> test
[info] Updating {file:/Users/wakita/Dropbox%20(smartnova)/doc/classes/cs1/lecture/}lecture...
[info] Resolving jline#jline;2.12.1 ...
[info] Done updating.
[info] Compiling 3 Scala sources to /Users/wakita/tmp/cs1f/scala-2.11/classes...
[info] Compiling 2 Scala sources to /Users/wakita/tmp/cs1f/scala-2.11/test-classes...
[info] Test:
[info] leapyear
[info] - should be true for 4で割り切れる年
[info] LX02ATest:
[info] leapyear
[info] - should be true for 4で割り切れる年
[info] leapyear
[info] - should be false for 4で割り切れない年
[info] leapyear
[info] - should be false to 100で割り切れる年
[info] Run completed in 525 milliseconds.
[info] Total number of tests run: 4
[info] Suites: completed 2, aborted 0
[info] Tests: succeeded 4, failed 0, canceled 0, ignored 0, pending 0
[info] All tests passed.
[success] Total time: 6 s, completed 2015/10/13 10:01:27
>
```

全テストをパス。完璧！

# と、喜んでいると、天の声

---

- ❖ 曰く「4で割り切れない年は平年」
- ❖ 「やべ、テストが甘い！追加しなくちゃ」



# 4で割り切れない年のテストを追加

```
lx02a-leapyear.scala (~/.Dropbox (smartn
-NERD_tree_1 lx02a-leapyear.s...
1 // このファイルについて
2 //   lecture/test/lx02a-leapyear.scala
3 //   UTF8
4
5 package cs1.lx02a
6
7 import org.scalatest._
8 import LX02A._
9
10 class Test extends FlatSpec with Matchers {
11
12   "leapyear" should "be true for 4で割り切れる年" in {
13     leapyear(2004) should be (true)
14     leapyear(2008) should be (true)
15   }
16
17   "leapyear" should "be false for 4で割り切れない年" in {
18     leapyear(2001) should be (false)
19     leapyear(2002) should be (false)
20     leapyear(2003) should be (false)
21   }
22 }
```

# 再度テストを実行

```
1. Default
> test
[info] Test:
[info] leapyear
[info] - should be true for 4で割り切れる年
[info] leapyear
[info] - should be false for 4で割り切れない年 *** FAILED ***
[info]   true was not false (lx02a-leapyear.scala:18)
[info] Run completed in 477 milliseconds.
[info] Total number of tests run: 2
[info] Suites: completed 1, aborted 0
[info] Tests: succeeded 1, failed 1, canceled 0, ignored 0, pending 0
[info] *** 1 TEST FAILED ***
[error] Failed tests:
[error]   cs1.lx02a.Test
[error] (test:test) sbt.TestsFailedException: Tests failed
[error] Total time: 1 s, completed
> 
```

leapyearのテストで問題発見

テスト (lx02a-leapyear.scala)の18行目を見て  
false が欲しい (should be false for ...) のに、  
実際は true じゃん (true was not false) ぶん。



# 再度テストを実行

```
1. Default
> test
[info] Test:
[info] leapyear
[info] - should be true for 4で割り切れる年
[info] leapyear
[info] - should be false for 4で割り切れない
[info] true was not false (lx02a-leapyear
[info] Run completed in 477 milliseconds.
[info] Total number of tests run: 2
[info] Suites: completed 1, aborted 0
[info] Tests: succeeded 1, failed 1, canceled 0, ignored 0, pending 0
[info] *** 1 TEST FAILED ***
[error] Failed tests:
[error]       cs1.lx02a.Test
[error] (test:test) sbt.TestsFailedException: Tests unsuccessful
[error] total time: 1 s, completed 2015/10/13 10:24:57
> |
```

一箇所コケたよ

コケたテストは cs1.lx02a.Test

残念

そこでテストコードの18行目を見ると、もちろんテストの内容は正しい


```
1 // このファイルについて
2 //   lecture/test/lx02a-leapyear.scala
3 //   UTF8
4
5 package cs1.lx02a
6
7 import org.scalatest._
8 import LX02A._
9
10 class Test extends FlatSpec with Matchers {
11
12   "leapyear" should "be true for 4で割り切れる年" in {
13     leapyear(2004) should be (true)
14     leapyear(2008) should be (true)
15   }
16
17   "leapyear" should "be false for 4で割り切れない年" in {
18     leapyear(2001) should be (false)
19     leapyear(2002) should be (false)
20     leapyear(2003) should be (false)
21   }
22 }
```



で、プログラムの問題を探す  
(までもなく、明らかに適当なのだが)

---

- ❖ 以下を修正して、`leapyear(2001) → false` となるようにすればよい。



The screenshot shows a code editor window with two tabs labeled 'lx02a-leapyear.s...'. The code is in Scala and includes comments in Japanese. The function definition for 'leapyear' is highlighted with a red hand-drawn rectangle. The code is as follows:

```
// ファイルについて
//   lecture/src/lx02a-leapyear.scala
//   UTF8

package cs1.lx02a

object Main {
  def leapyear(y: Int) = {
    false
  }
}
```

- ❖ (半端に) ずる賢い人は `leapyear(y) = { false }` に変更

19行目は ok だが,

今度はさっきは成功していた13行目が. . .

```
1. Default
> test
[info] Compiling 1 Scala source to /Users/wakita/tmp/cs1f/scala-2.11/classes...
[info] Test:
[info] leapyear
[info] - should be true for 4で割り切れる年 *** FAILED ***
[info]   false was not true (lx02a-leapyear.scala:13)
[info] leapyear
[info] - should be false for 4で割り切れない年
[info] Run completed in 502 milliseconds.
[info] Total number of tests run: 2
[info] Suites: completed 1, aborted 0
[info] Tests: succeeded 1, failed 1, canceled 0
[info] *** 1 TEST FAILED ***
[error] Failed tests:
[error]       cs1.lx02a.Test
[error] (test:test) sbt.TestsFailedException: Tests unsuccessful
[error] Total time: 1 s, completed 2015/10/13 10:33:37
> 
```

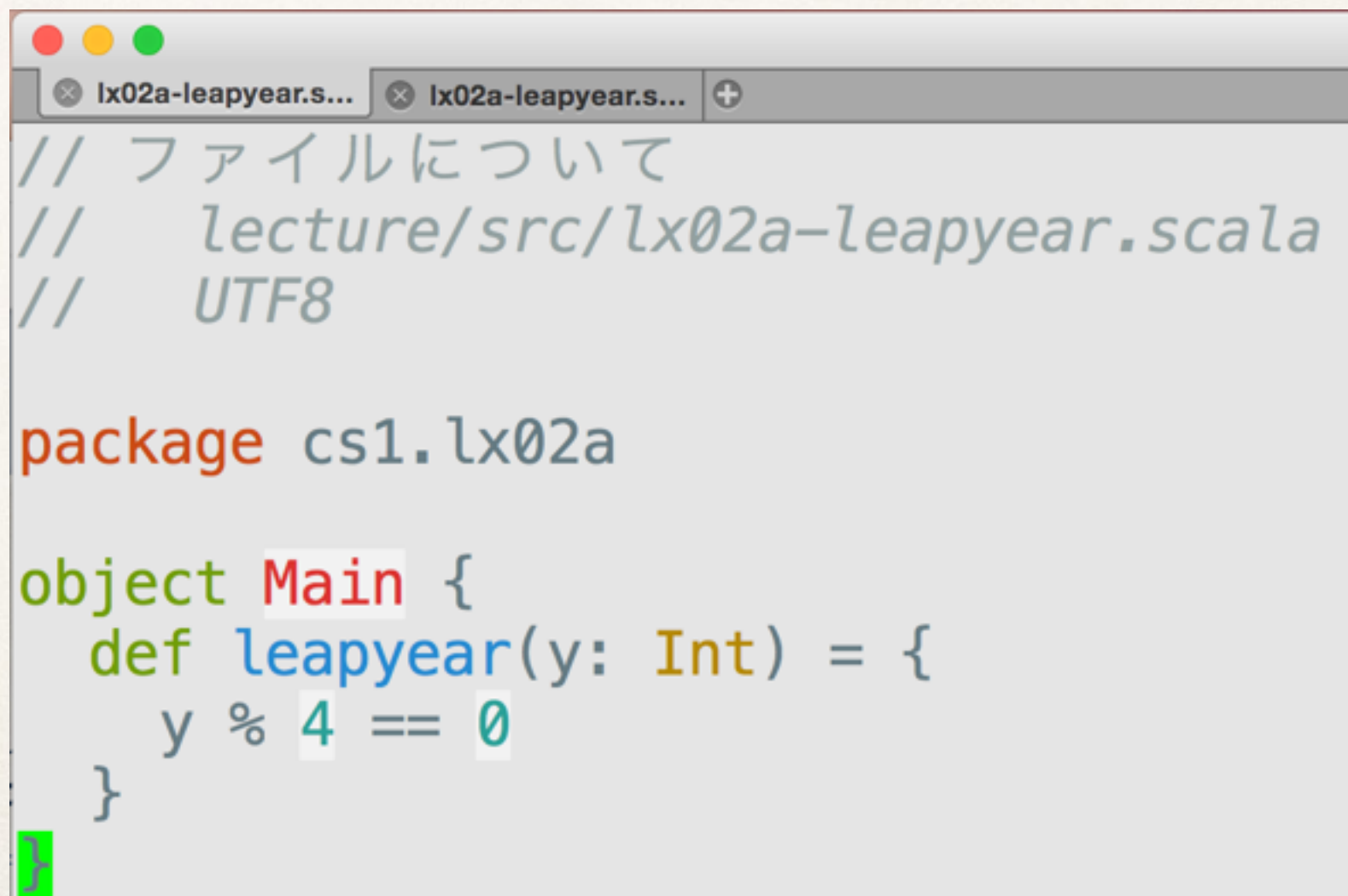
leapyear(2004) should be (true)



もう少し真面目に対応するか  
4で割り切れれば閏年なんですよ？

---

第一の条件：西暦年が4で割り切れる年は閏年



```
// ファイルについて
//   lecture/src/lx02a-leapyear.scala
//   UTF8

package cs1.lx02a

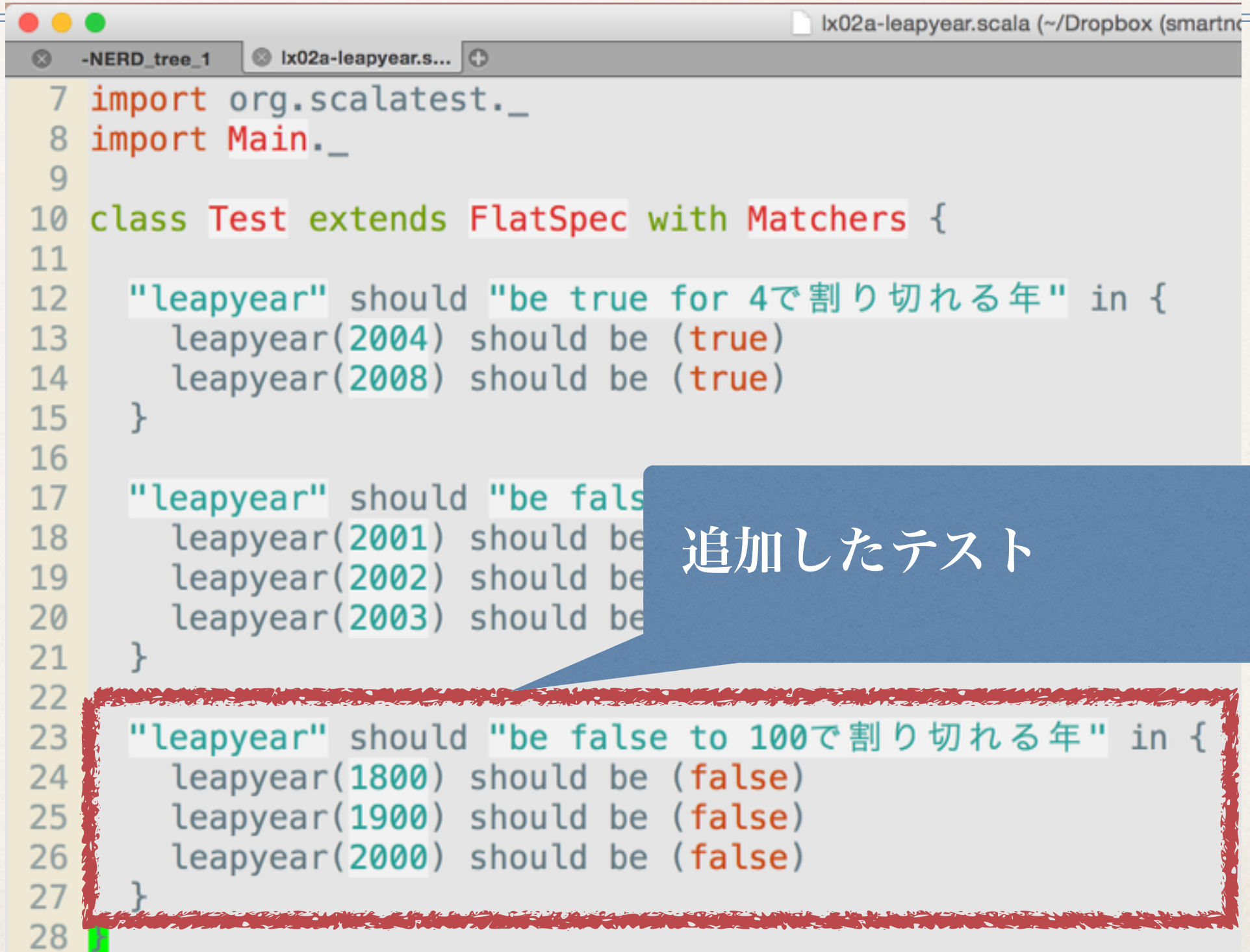
object Main {
  def leapyear(y: Int) = {
    y % 4 == 0
  }
}
```

# やった～！すべてパス

```
1. Default
> test
[info] Test:
[info] leapyear
[info] - should be true for 4で割り切れる年
[info] leapyear
[info] - should be false for 4で割り切れない年
[info] Run completed in 460 milliseconds.
[info] Total number of tests run: 2
[info] Suites: completed 1, aborted 0
[info] Tests: succeeded 2, failed 0, canceled 0, ignored 0, pending 0
[info] All tests passed.
[success] Total time: 1 s, completed 2015/10/13 10:37:03
> █
```



# 調子にのって、 二番目のテストを追加



```
7 import org.scalatest._
8 import Main._
9
10 class Test extends FlatSpec with Matchers {
11
12   "leapyear" should "be true for 4で割り切れる年" in {
13     leapyear(2004) should be (true)
14     leapyear(2008) should be (true)
15   }
16
17   "leapyear" should "be false for 4で割り切れない年" in {
18     leapyear(2001) should be (false)
19     leapyear(2002) should be (false)
20     leapyear(2003) should be (false)
21   }
22
23   "leapyear" should "be false to 100で割り切れる年" in {
24     leapyear(1800) should be (false)
25     leapyear(1900) should be (false)
26     leapyear(2000) should be (false)
27   }
28 }
```

追加したテスト

# 三度テストを実行

```
1. Default
> test
[info] Test:
[info] leapyear
[info] - should be true for 4で割り切れる年
[info] leapyear
[info] - should be false for 4で割り切れない年
[info] leapyear
[info] - should be false to 100で割り切れる年 *** FAILED ***
[info]   true was not false (lx02a-leapyear.scala:24)
[info] Run completed in 480 milliseconds.
[info] Total number of tests run: 3
[info] Suites: completed 1, aborted 0
[info] Tests: succeeded 2, failed 1, canceled 0, ignored 0, pending 0
[info] *** 1 TEST FAILED ***
[error] Failed tests:
[error]   cs1.lx02a.Test
[error] (test:test) sbt.TestsFailedException: Tests failed
[error] Total time: 1 s, completed 2015-01-01 10:00:00
> 
```

もちろんこける (2つ成功, 1つ失敗)

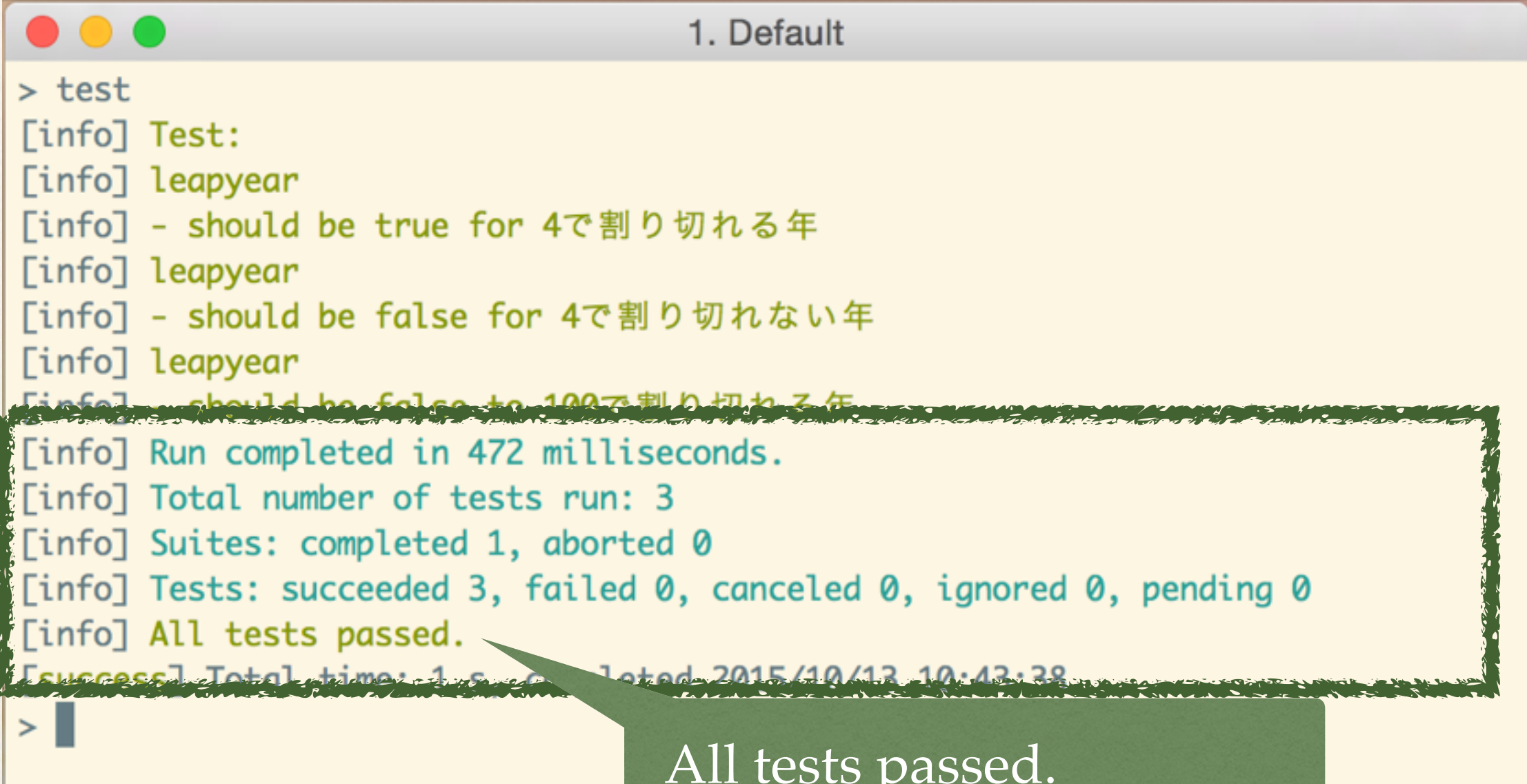


# テストにあわせて修正

---

```
leapyear.scala  ex01a-leapyear....  
object ex1a {  
  def leapyear(y: Int) = {  
    !(y % 100 == 0) &&  
    y % 4 == 0  
  }  
}
```

# 4度目のテスト



```
> test
[info] Test:
[info] leapyear
[info] - should be true for 4で割り切れる年
[info] leapyear
[info] - should be false for 4で割り切れない年
[info] leapyear
[info] - should be false for 100で割り切れる年
[info] Run completed in 472 milliseconds.
[info] Total number of tests run: 3
[info] Suites: completed 1, aborted 0
[info] Tests: succeeded 3, failed 0, canceled 0, ignored 0, pending 0
[info] All tests passed.
[success] Total time: 1 s, completed 2015/10/13 10:43:38

> █
```

All tests passed.  
緑の字が目によしいぜ！



# 天の声 いやいや、まだ駄目でしょ

---

- ❖ 曰く「ただし、西暦年が400で割り切れる年は閏年」
- ❖ ということは、2000年とか1600年は閏年？

# あとは任せた

---

- ❖ 課題の内容は別途詳述します。自分で閏年のプログラムを完成して下さい。
- ❖ 注意：今日の課題はほかにもあります。



# プログラム開発環境

---

# プログラム 開発環境

テキストエディタ

プログラムとテストコードを編集

作業内容

- コードの修正
- ファイルの保存
- 作業中はエディタのウィンドウは開きっぱなし

~  
<c/classes/cs1/src/main/scala/ex01a-leapyear.scala [utf-8-unix]

Takamiishi.jpg

```
1. Default
[info] leapyear
[info] - should be false for 4で割り切れない年
[info] leapyear
[info] - should be false to 100で割り切れる年
[info] Run completed in 835 milliseconds.
[info] Total number of tests run: 3
[info] Suites: completed 1, aborted 0
[info] Tests: succeeded 3, failed 0, canceled 0, i
[info] All tests passed.
[success] Total time: 1 s, completed 2014/10/07 12
> 
```

ターミナル

sbt を動かしっぱなし

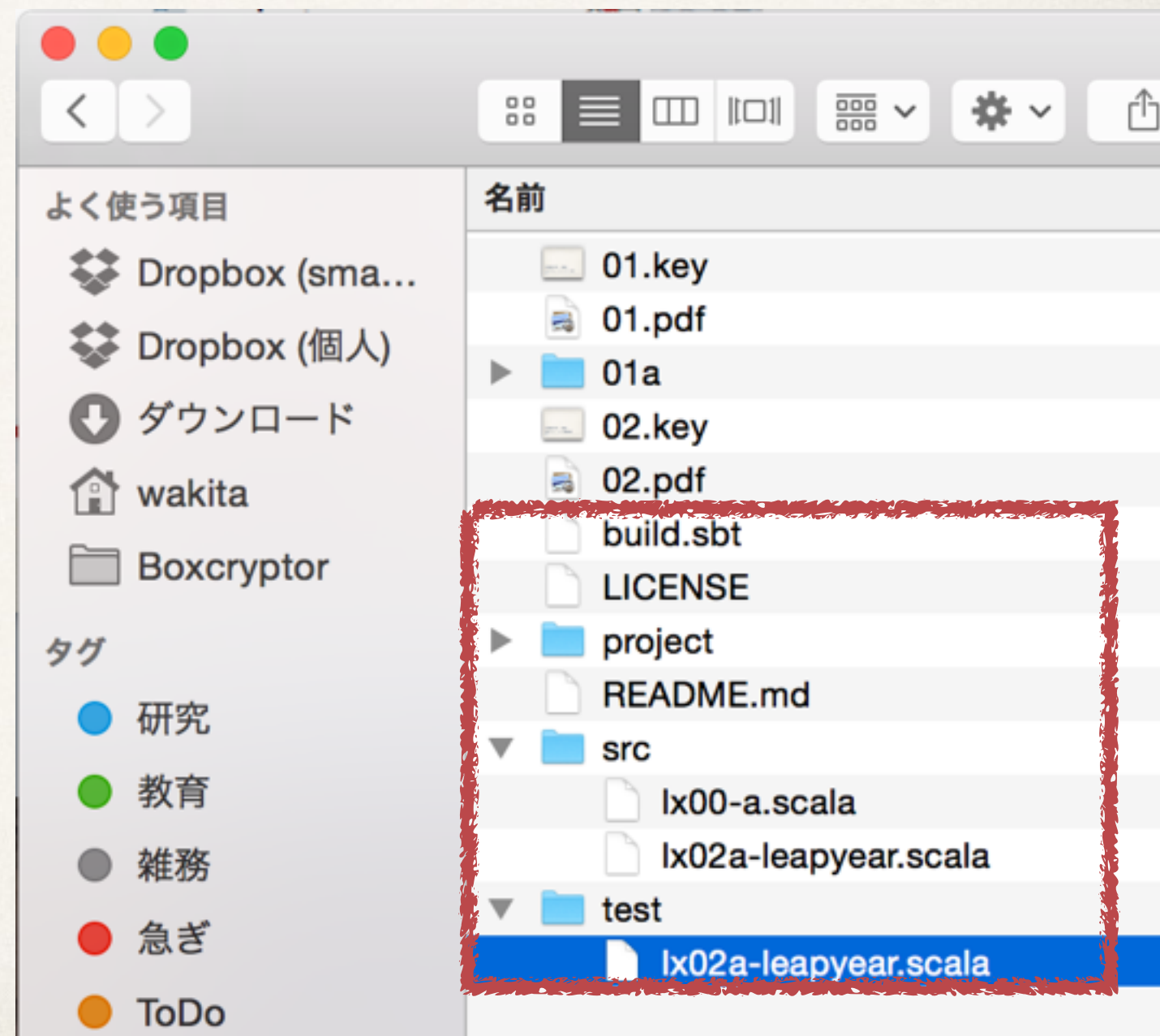
ときどき “test” を実行

“~test~”で継続テストをするのもよい



# フォルダの構成

- ❖ build.sbt: sbt の設定ファイル
- ❖ src/: プログラムの置き場所
- ❖ test/: テストコードの置き場所
- ❖ project: sbt が勝手に作る. 気にしない.



# cs1/build.sbt の主な内容

記述には一級の正確さが求められます

---

```
scalaVersion := "2.11.7"
```

```
scalacOptions ++= Seq("-optimize", "-feature", "-unchecked", "-  
deprecation")
```

```
libraryDependencies += "org.scalatest" % "scalatest_2.11" % "2.2.5" % "test"
```



# 課題1：演習時間内にTAの確認を受けて下さい

---

- ❖ 課題1についてTAの確認をもらうまでは、この課題に着手してはいけません。
- ❖ 授業で説明を受けた閏年のプログラムをテスト駆動方式にしたがって完成させなさい。  
ファイル:{src,test}/lx02a-leapyear.scala
- ❖ くれぐれも以下の手順を踏むこと
  - ❖ すべての仕様に合致するテスト(test/lx02a-leapyear.scala)を作成すること。不安な人は、テストの内容をTAに確認してもらって下さい。
  - ❖ 失敗したテストを順次、潰す要領で徐々にプログラム(src/lx02a-leapyear.scala)を完成すること。
- ❖ 完成したら、課題2に取り組む前に TA の確認を受けること。



# 宿題：パズルを解くプログラムを完成させなさい

---

- ❖ 右のパズルを自動的に解くプログラムをテスト駆動開発方式で作成しなさい。

(科学雑誌Newton@facebook)

- ❖ ファイル

{src,test}/lx02b-puzzle.scala

この紙の上には、  
1 という数字が  個、  
2 という数字が  個、  
3 という数字が  個、  
1 から 3 まで以外の数字が  個書いてある。