

# 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

# ToDo

---

- ❖ build.sbt の更新
- ❖ Scalatest を用いたテストの記述方法
- ❖ build.sbt の記述方法



# 講義資料の入手方法

---

- ❖ 先週、作成した lecture ディレクトリに移動してから  
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：ひとまず簡単なコードを作成 バグを含んでいてよい

---

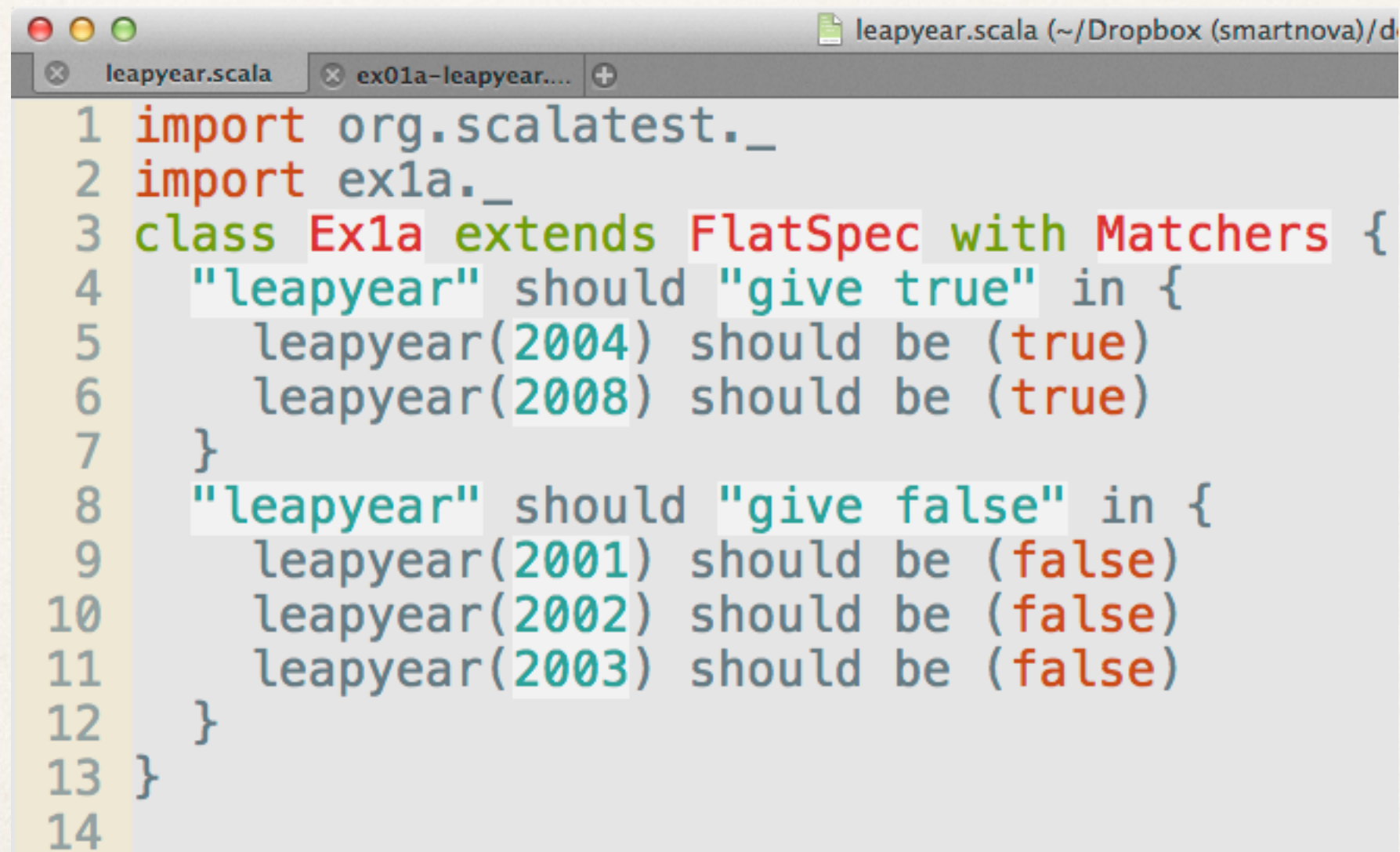
```
// src/main/scala/ex01a-leapyear.scala
```

```
object ex1a {  
  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

---

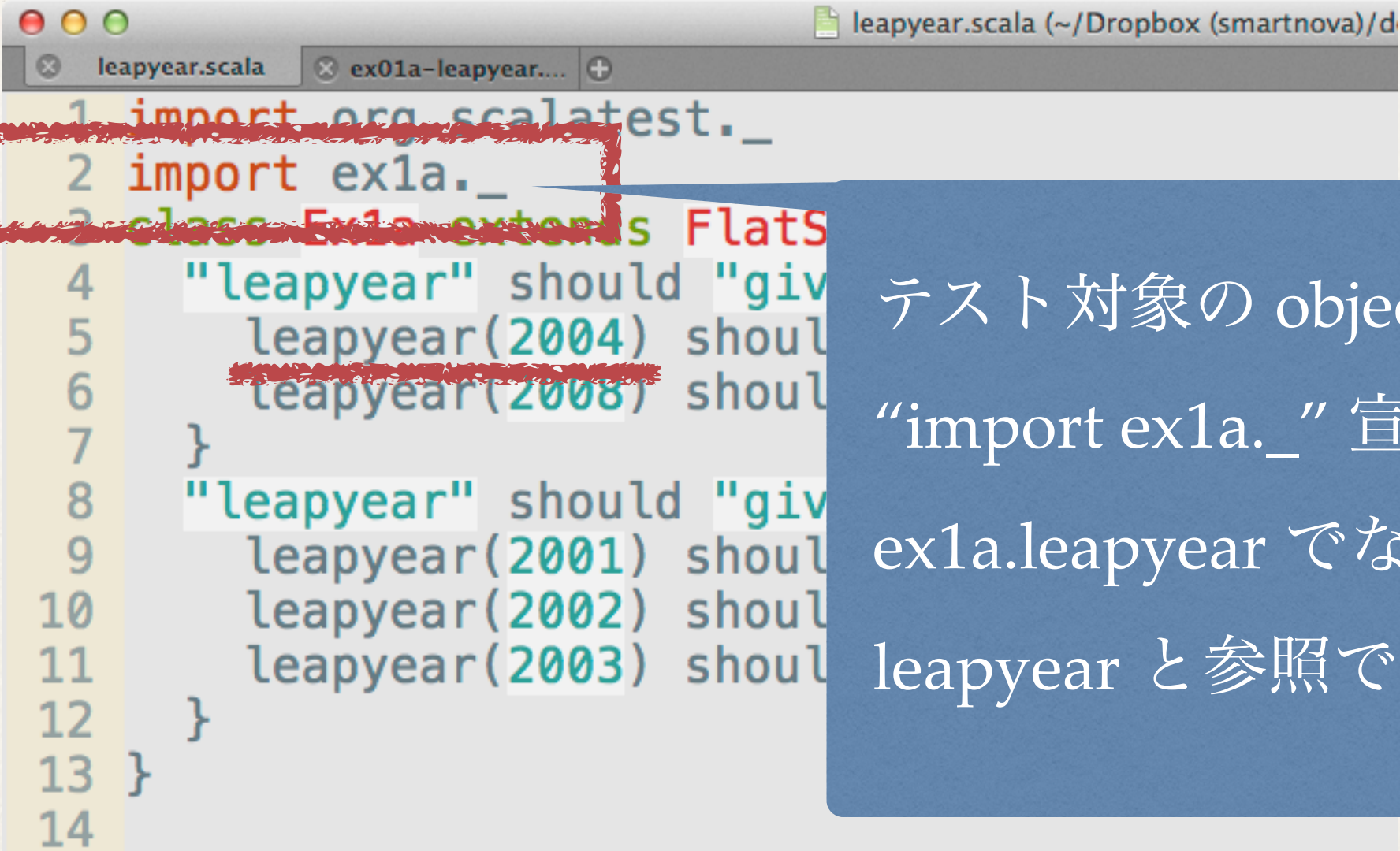


```
leapyear.scala (~/Dropbox (smartnova)/d
leapyear.scala  ex01a-leapyear...
1 import org.scalatest._
2 import ex1a._
3 class Ex1a extends FlatSpec with Matchers {
4   "leapyear" should "give true" in {
5     leapyear(2004) should be (true)
6     leapyear(2008) should be (true)
7   }
8   "leapyear" should "give false" in {
9     leapyear(2001) should be (false)
10    leapyear(2002) should be (false)
11    leapyear(2003) should be (false)
12  }
13 }
14
```



# 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

---



```
leapyear.scala (~/Dropbox (smartnova)/d
leapyear.scala  ex01a-leapyear...
1 import org.scalatest._
2 import ex1a._
3 class Ex1a extends FlatSpec
4   "leapyear" should "give"
5     leapyear(2004) should
6     leapyear(2008) should
7   }
8   "leapyear" should "give"
9     leapyear(2001) should
10    leapyear(2002) should
11    leapyear(2003) should
12  }
13 }
14
```

テスト対象の object の名前  
“import ex1a.\_” 宣言により,  
ex1a.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] Compiling 1 Scala source to /Users/wakita/Dropbox (smartnova)/doc/classes
/cs1/target/scala-2.10/test-classes...
[info] Ex1a:
[info] leapyear
[info] - should give true
[info] leapyear
[info] - should give false *** FAILED ***
[info]   true was not false (leapyear.scala:9)
[info] Run completed in 825 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]       Ex1a
[error] (test:test) sbt.TestsFailedException: Tests unsuccessful
[error] Total time: 2 s, completed 2014/10/07 11:08:31
> █
```



# sbtコマンドを起動してテストを実行

## Scala Build Tool

```
1. Default
> test
[info] Compiling 1 Scala source to /Users/wakita/Dropbox (smartnova)/doc/classes
/cs1/target/scala-2.10/test-classes...
[info] Ex1a:
[info] leapyear
[info] - should give true
[info] leapyear
[info] - should give false *** FAILED ***
[info]   true was not false (leapyear.scala:9)
[info] Run completed in 825 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]   Ex1a
[error] (test:test) sbt.TestsFailedException:
[error] Total time: 2 s, completed 2014/10/
>
```

一箇所テストがこけたよ

# sbtコマンドを起動してテストを実行

## Scala Build Tool

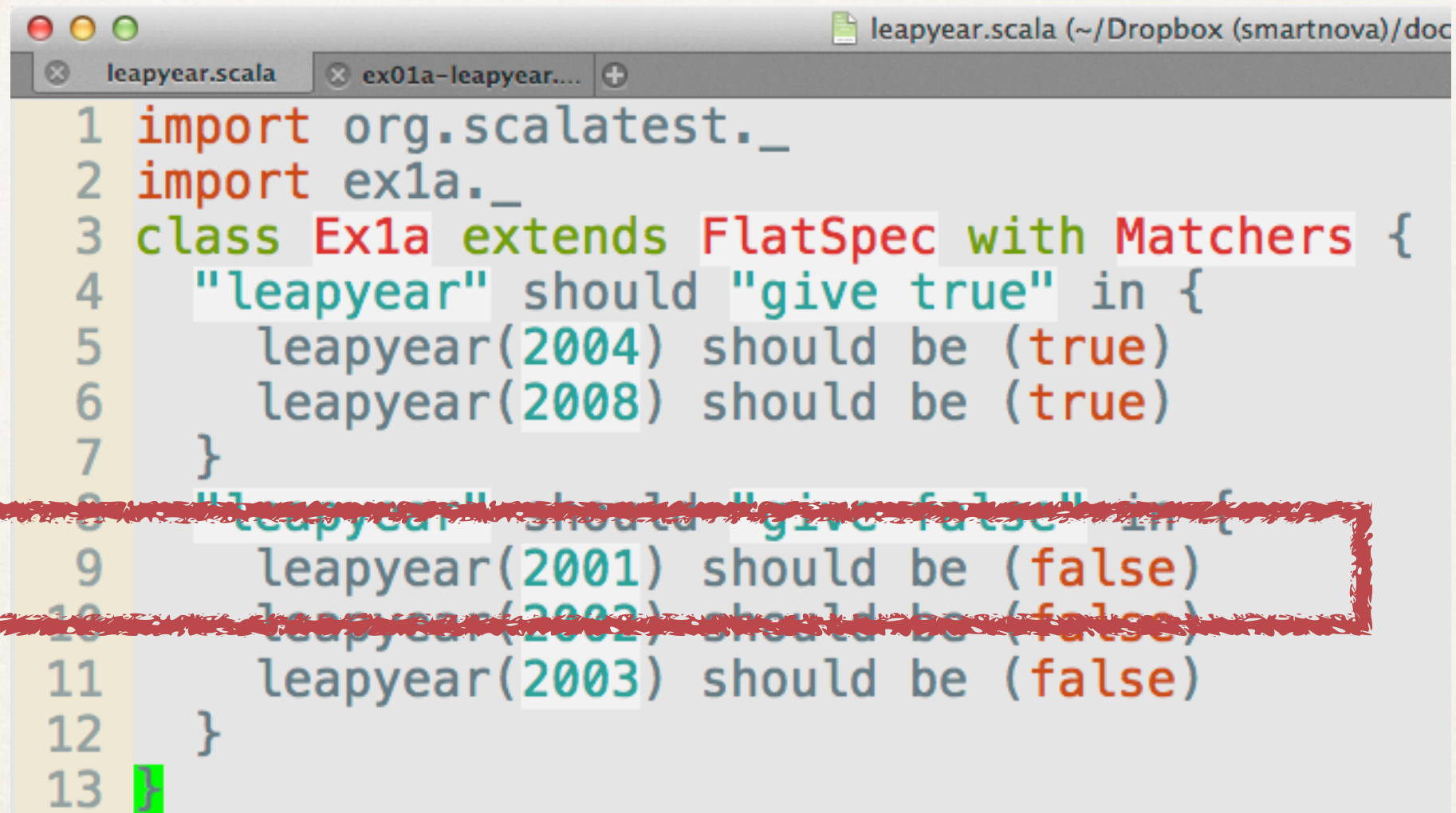
```
1. Default
> test
[info] Compiling 1 Scala source to /Users/wakita/Dropbox (smartnova)/doc/classes
/cs1/target/scala-2.10/test-classes...
[info] Ex1a:
[info] leapyear
[info] - should give true
[info] leapyear
[info] - should give false *** FAILED ***
[info]   true was not false (leapyear.scala:9)
[info] Run completed in 825 milliseconds.
[info] Total number of tests: 1
[info] Suites: completed 1, aborted 0
[info] Tests: succeeded 1, failed 1, total 2
[info] *** 1 TEST FAILED ***
[error] Failed tests:
[error]   Ex1a
[error] (test:test) sbt.TestsFailedException: Tests failed
[error] Total time: 2 s, completed 2015/01/14 11:11:11
>
```

テスト (leapyear.scala)の9行目を見て  
ここでfalse が欲しかった (should give  
false) のに、結果は true だったよ。



そこでテストコードの9行目を見ると、  
もちろんテストは正しい

---

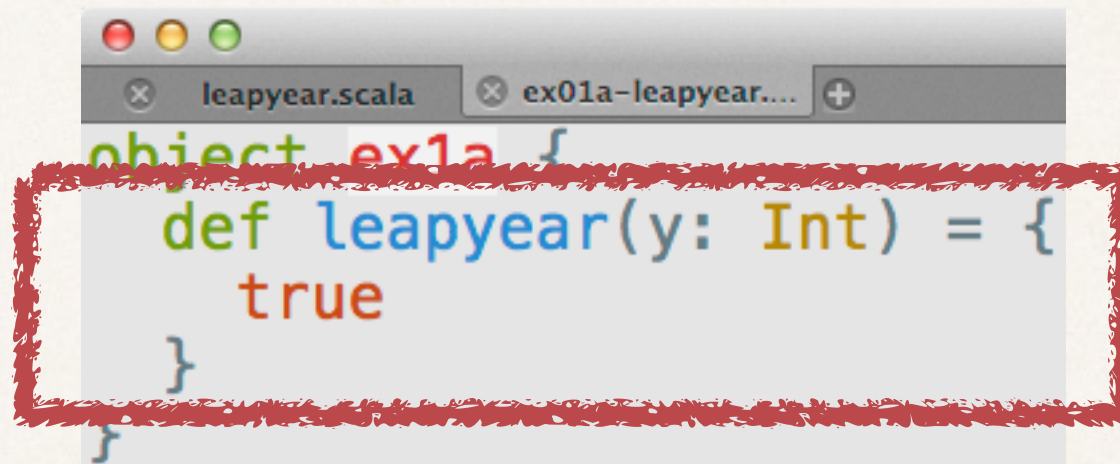


```
leapyear.scala (~/Dropbox (smartnova)/doc
leapyear.scala ex01a-leapyear....
1 import org.scalatest._
2 import ex1a._
3 class Ex1a extends FlatSpec with Matchers {
4   "leapyear" should "give true" in {
5     leapyear(2004) should be (true)
6     leapyear(2008) should be (true)
7   }
8   "leapyear" should "give false" in {
9     leapyear(2001) should be (false)
10    leapyear(2002) should be (false)
11    leapyear(2003) should be (false)
12  }
13 }
```

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.



A screenshot of a Scala IDE window. The window has two tabs: 'leapyear.scala' and 'ex01a-leapyear...'. The code in the editor is as follows:

```
object ex1a {  
  def leapyear(y: Int) = {  
    true  
  }  
}
```

The function definition 'def leapyear(y: Int) = { true }' is highlighted with a red hand-drawn rectangular box.

- ❖ 賢い人は leapyear(y) = { false } に変更



# 9行目は ok だが, 今度は5行目が

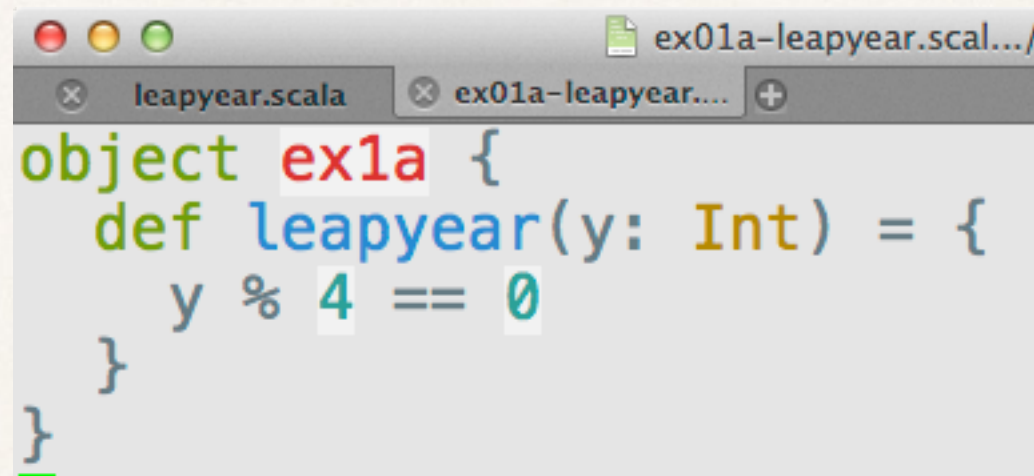
```
1. Default
> test
[info] Compiling 1 Scala source to /Users/wakita/Dropbox (smartnova)/doc/classes
/cs1/target/scala-2.10/classes...
[info] Ex1a:
[info] leapyear
[info] - should give true *** FAILED ***
[info]   false was not true (leapyear.scala:5)
[info] leapyear
[info] - should give false
[info] Run completed in 943 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]       Ex1a
[error] (test:test) sbt.TestsFailedException: Tests unsuccessful
[error] Total time: 1 s, completed 2014/10/07 11:24:58
> 
```

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

A screenshot of a Scala IDE window. The window has two tabs: 'leapyear.scala' and 'ex01a-leapyear...'. The code in the editor is as follows:

```
object ex1a {  
  def leapyear(y: Int) = {  
    y % 4 == 0  
  }  
}
```



# やった～！すべてパス

```
1. Default
> test
[info] Ex1a:
[info] leapyear
[info] - should give true
[info] leapyear
[info] - should give false
[info] Run completed in 816 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 2014/10/07 11:40:20
>
```

# と思うのもつかのま

---

❖ 天の声，曰く

❖ 「ただし、西暦年が100で割り切れる年は平年」

❖ 「やべ，テストが甘い！追加しなくちゃ」



# テストの追加

```
1
2
3
4
5  "leapyear" should "be true for 4で割り切れる年" in {
6    leapyear(2004) should be (true)
7    leapyear(2008) should be (true)
8  }
9
10 "leapyear" should "be false for 4で割り切れない年" in {
11   leapyear(2001) should be (false)
12   leapyear(2002) should be (false)
13   leapyear(2003) should be (false)
14 }
15
16 "leapyear" should "be false to 100で割り切れる年" in {
17   leapyear(1800) should be (false)
18   leapyear(1900) should be (false)
19   leapyear(2000) should be (false)
20 }
21 }
22
```

追加したテスト

# 三度テストを実行

```
> test
[info] Ex1a:
[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 (leapyear.scala:17)
[info] Run completed in 828 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]   Ex1a
[error] (test:test) sbt.TestsFailedException: Tests unsuccessful
[error] Total time: 1 s, completed 2014/10/07 11:52:58
```

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] Ex1a:
[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 805 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, 50 ms
```

All tests passed. 嬉しい！



# いやいや、まだ駄目でしょ

---

- ❖ 「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

ターミナル

sbt を動かしっぱなし

ときどき “test” を実行

```
ex01a-leapyear.scala (~/.../cs1/src/main/scala) - VIM
leapyear.scala  ex01a-leapyear...
def leapyear(y: Int) = {
  !(y % 100 == 0) &&
  y % 4 == 0
}
```

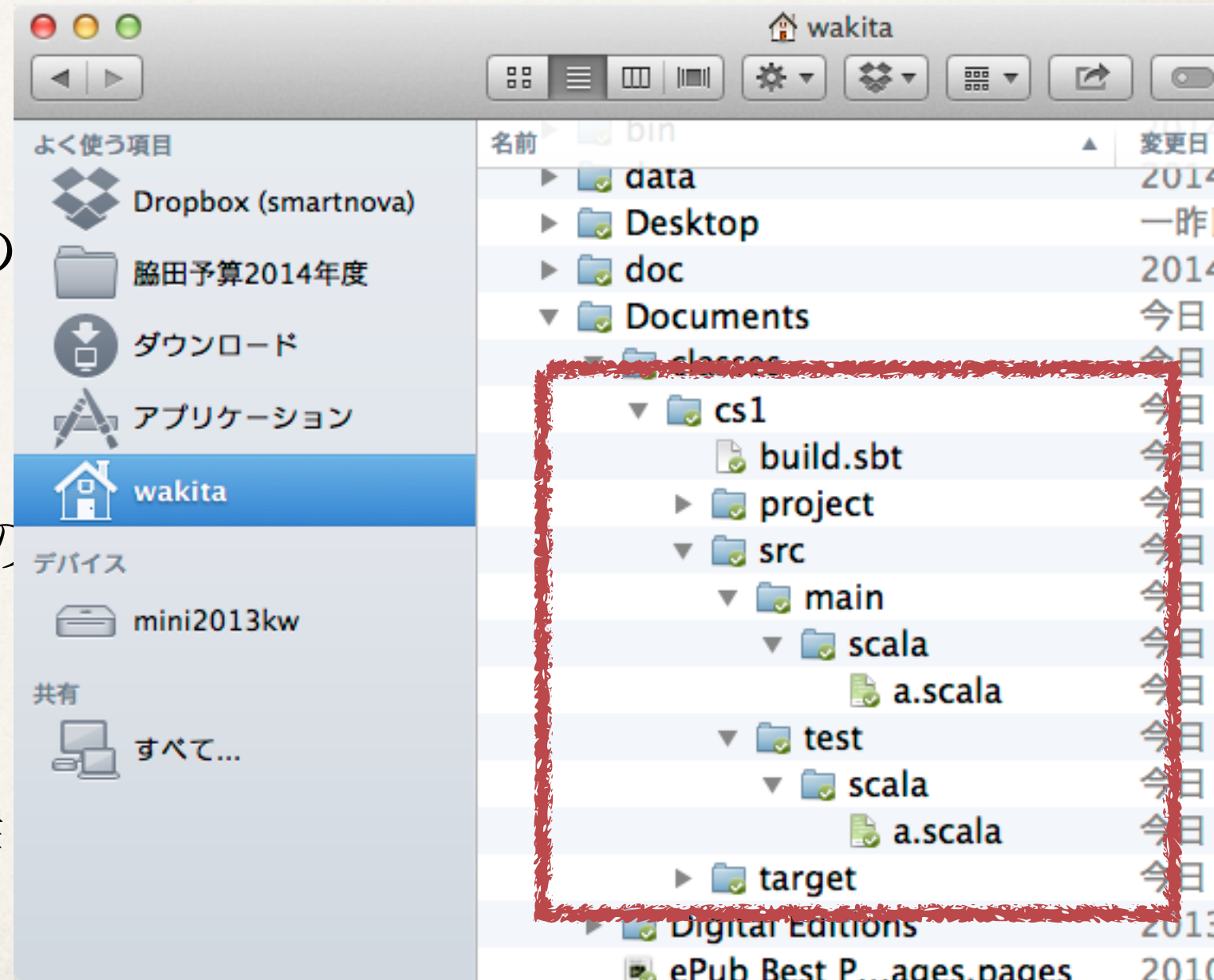
<c/classes/cs1/src/main/scala/ex01a-leapyear.scala [utf-8-unix]

```
1. Default
[info] leapyear
[info] - should be false for 4で割
[info] leapyear
[info] - should be false to 100で
[info] Run completed in 835 millis
[info] Total number of tests run:
[info] Suites: completed 1, aborted
[info] Tests: succeeded 3, failed
[info] All tests passed.
[success] Total time: 1 s, comple
> 
```



# フォルダの構成

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



# cs1/build.sbt の内容

## 正確に記入すること

---

```
scalaVersion := "2.10.0"
```

```
libraryDependencies += "org.scalatest" %% "scalatest_2.10" % "2.0" % "test"
```

```
scalacOptions += "-optimise"
```



# ターミナルを開き授業用のフォルダ を作成, フォルダへ移動

---

```
mkdir -p Documents/classes  
cd Documents/classes
```

# プログラム用フォルダの作成

---

```
mkdir -p cs1/src/main/scala
```



# テスト用フォルダの作成

---

```
mkdir -p cs1/src/test/scala
```

# sbtの実行

---

`~/aotani/local/sbt/bin/sbt`



# 課題1

---

- ❖ 前述の要領で作業用フォルダを作成すること
- ❖ きちんとできたら，次の課題に移る前に TA の確認をもらうこと

# 課題2

---

- ❖ 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/{main,test}/scala/ex01a-leapyear.scala`
- ❖ くれぐれもプログラムを完成させてから，テストを追加しないこと。
- ❖ 完成したら，課題3に取り組む前に TA の確認を受けること。



## 課題3：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方式で作成しなさい。

(科学雑誌Newton@facebook)

- ❖ ファイル

src/{main,test}/scala/ex01b-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。