

sbt

Contents

Preface	4
sbt	4
.	4
Mac sbt	4
.	4
.	5
.	5
Windows sbt	5
Windows	5
.	5
.	5
Linux sbt	5
.	5
RPM DEB	6
Gentoo	6
.	6
sbt	6
Unix	6
Windows	7
Hello, World	7
.	8
.	8

sbt	9
.	9
.	9
.	9
sbt	10
.	10
.	10
.	10
.	10
.	11
.	11
.	11
Tab	12
.	12
.sbt	12
.sbt vs .scala	13
?	13
build.sbt	13
.	14
(Keys)	14
tasks settings	15
sbt Keys	15
build.sbt	16
.	16
Scope	16
Key	16
Scope	17
Scope	18
.	18
sbt scoped key	18
scoped key	18

scope	19
scope	20
scope	21
.	22
:	22
: += +=	22
key	23
:= +=	24
.	24
.	25
.	25
.	28
.	28
.	28
root	29
.	30
.	30
.	30
.	30
.	31
.	31
.	32
.	32
.	33
.	33
.	33
.	34
.scala	34
sbt	34
.scala	35
build.sbt Build.scala	35

.scala	37
.	37
:	37
.	37
sbt:	38
.	38

Preface

sbt , sbt , ,

sbt

!

, .sbt ,scopes,

,

sbt !

sbt

sbt , :

- sbt
- hello world

—
—

- sbt sbt
- .sbt

, Jar Shell , Mac,Windows,Linux,

sbt , ,HTTP ,JVM

Mac sbt

: ,

Macports

```
$ port install sbt
```

Homebrew

```
$ brew install sbt
```

[ZIP](#) [TGZ](#)

Windows [sbt](#)

Windows

[msi](#)

[ZIP](#) [TGZ](#)

Linux [sbt](#)

[ZIP](#) [TGZ](#)

RPM DEB

:

- [RPM](#)
- [DEB](#)

: [sbt-launcher-package](#)

Gentoo

sbt ebuild sbt [ebuilds](#) ebuilds sbt:

```
$ mkdir -p /usr/local/portage && cd /usr/local/portage
$ git clone git://github.com/whiter4bbit/overlays.git
$ echo "PORTDIR_OVERLAY=$PORTDIR_OVERLAY /usr/local/portage/overlays" >> /etc/make.conf
$ emerge sbt-bin
```

: ebuild

sbt

[sbt-launch.jar](#),

Unix

[sbt-launch.jar](#) ~/bin

jar, ~/bin/sbt :

```
SBT_OPTS="-Xms512M -Xmx1536M -Xss1M -XX:+CMSClassUnloadingEnabled -XX:MaxPermSize=256M"
java $SBT_OPTS -jar ` $0`/sbt-launch.jar "$@"
```

:

```
$ chmod u+x ~/bin/sbt
```

Windows

Windows Cygwin , batch path , sbt
sbt , , , JVM

Non-Cygwin Windows Cygwin , sbt.bat batch :

```
set SCRIPT_DIR=%~dp0
java -Xms512M -Xmx1536M -Xss1M -XX:+CMSClassUnloadingEnabled -XX:MaxPermSize=256M -jar "%SCRIP
```

[sbt-launch.jar](#) sbt.bat

Cygwin **Windows** Cygwin Windows , bash ~/bin/sbt:

```
SBT_OPTS="-Xms512M -Xmx1536M -Xss1M -XX:+CMSClassUnloadingEnabled -XX:MaxPermSize=256M"
java $SBT_OPTS -jar sbt-launch.jar "$@"
```

[sbt-launch.jar](#) sbt-launch.jar, cygpath :

```
$ chmod u+x ~/bin/sbt
```

Cygwin **Ansi** Cygwin Ansi (Ansi stty), bash
~/bin/sbt:

```
SBT_OPTS="-Xms512M -Xmx1536M -Xss1M -XX:+CMSClassUnloadingEnabled -XX:MaxPermSize=256M"
stty -icanon min 1 -echo > /dev/null 2>&1
java -Djline.terminal=jline.UnixTerminal -Dsbt.cygwin=true $SBT_OPTS -jar sbt-launch.jar "$@"
stty icanon echo > /dev/null 2>&1
```

[sbt-launch.jar](#) sbt-launch.jar, cygpath :

```
$ chmod u+x ~/bin/sbt
```

(backspace) scala , (erase character), stty cygwin
(mintty), -> ,“ ^H” cygwin ^H

: [pull request](#)

Hello, World

[sbt](#)

```

    sbt hello , hw.scala:

object Hi {
  def main(args: Array[String]) = println("Hi!")
}

hello sbt, run sbt Linux OS X :

$ mkdir hello
$ cd hello
$ echo 'object Hi { def main(args: Array[String]) = println("Hi!") }' > hw.scala
$ sbt
...
> run
...
Hi!

,sbt sbt :

•
• src/main/scala src/main/java
• src/test/scala src/test/java
• src/main/resources src/test/resources
• lib jar

,sbt Scala

sbt run sbt console Scala REPL sbt console class-
path, Scala

build.sbt
, hello , hello/build.sbt :

name := "hello"

version := "1.0"

scalaVersion := "2.10.3"

, .sbt build.sbt

jar , build.sbt name version

```



```

sbt

    hello/project/build.properties      sbt      ,      0.13.6:

sbt.version=0.13.6

sbt      release      99%      project/build.properties      sbt

sbt      Hello, World

sbt      ,“ ”      ,      Hello, World      hello ,      hello/build.sbt
hello/hw.scala, hello

    hello/hw.scala      ,      ,

sbt      Maven      (      ):

src/
  main/
    resources/
      <files to include in main jar here>
    scala/
      <main Scala sources>
    java/
      <main Java sources>
  test/
    resources
      <files to include in test jar here>
    scala/
      <test Scala sources>
    java/
      <test Java sources>

src/      ,

```

```

sbt

      build.sbt  sbt  project
project      .scala  ,      .sbt      .scala

build.sbt
project/
  Build.scala

      project/      .sbt  ,      .sbt      ,

      (  classes,  jars,  ,caches  )  target

.gitignore (      ) :

target/

:      / (      )      / (      target/  project/target/ )

      sbt      sbt      Hello, World

      sbt      :

$ sbt

      sbt      (  tab      )
, sbt      compile:

> compile

      compile,
      run
      exit  Ctrl+D (Unix)  Ctrl+Z (Windows)

```

```

sbt, sbt , sbt :

$ sbt clean compile "testOnly TestA TestB"

, testOnly TestA TestB (clean, compile, testOnly)

- - , sbt
~ , :

> ~ compile

~

sbt
clean
( target )
compile
( src/main/scala src/main/java )
test

console
classpath Scala :quit, Ctrl+D (Unix), Ctrl+Z (Windows)
sbt
run < >*
sbt main class
package
src/main/resources src/main/scala src/main/java class jar

```

help < >

,

reload

(build.sbt, project/.scala, project/.sbt)

Tab

tab sbt , tab ,

, sbt :

!

!!

!:

!n

n

!n

n , !:

!-n

n

!string

string

!?string

string

.sbt

sbt , “ ” build.sbt sbt

.sbt vs .scala

```
sbt      .sbt      ,      project/      .scala
      .sbt .scala      .sbt      ,      .scala      (      )      .scala
```

?

```
sbt      ,      (immutable map)(      )
, name key, value
      sbt map
      ,      Setting[T]      ,T      (value)      Setting      (map)      ,
      value (      , map      map      ,      map )
build.sbt ,      Setting[String]:
```

```
name := "hello"
```

```
Setting[String]      (      ) name key, value "hello" map      map
sbt map
      map,sbt      ,      key      ,      value      key,      key      , sbt
Settings      ,      map
:      Setting[T]      ,Setting[T]      sbt      map      ,T      value
```

build.sbt

```
build.sbt Seq[Setting[_]];      Scala      ,      (sequence)
.sbt Seq(      ),      ,      .scala
:
```

```
name := "hello"
```

```
version := "1.0"
```

```
scalaVersion := "2.10.3"
```

```
Setting      Scala      build.sbt      ,      ,      Scala      val,lazy
val,def build.sbt      object class      project/      Scala
,name,version scalaVersion (keys) (key) SettingKey[T],TaskKey[T]
InputKey[T] ,T value key
(Keys) Setting[T] :=      Java      :
```

```

name.:=("hello")

,Scala    name := "hello" ( Scala ,
)
(key)name := Setting, Setting[String] String name
SettingKey[String] , Setting[String] sbt map name
, "hello"
value, :

name := 42 //

```

```

build.sbt:

// ,
name := "hello"
version := "1.0"
scalaVersion := "2.10.3"

sbt ,
.sbt Scala , Scala

```

(Keys)

(Types) key:

- SettingKey[T]: key value(,)
- TaskKey[T]: key task value, ,
- InputKey[T]: task [Input Tasks](#)

```

Keys keys Keys build.sbt import sbt.Keys._, name
sbt.Keys.name

```

```

Keys :settingKey,taskKey inputKey keys key value
key val , task hello key,

```

```

lazy val hello = taskKey[Unit](" task ")

```

```

.sbt (settings), vals defs (settings) vals
defs (settings)

: , lazy val val

```

```

Task vs Setting keys TaskKey[T] task Tasks compile
package Unit(Unit Scala void), task , package
TaskKey[File] task, jar
task, sbt compile,sbt task
sbt map (setting) , name; task , compile -
,
key task (setting) ,“taskiness” ( ) key (prop-
erty), (value)

```

tasks settings

```

:= setting task setting, value task, task

, hello task:

hello := { println("Hello!") }

settings ,

name := "hello"

```

```

Tasks Settings , task key Setting setting key
Setting taskKey := 42 Setting[Task[T]] settingKey := 42
Setting[T] ;task key T value
T Task[T] : setting task, setting ,

```

sbt Keys

```

sbt , task name task compile compile task compile
task key

setting key name task key name,setting key (value) task
key name task (value); show <task name> <task name>
task key name camelCase, name Scala

key , sbt inspect <keyname> inspect , setting
value setting

```

build.sbt

```
build.sbt ;
:

import sbt._
import Process._
import Keys._

( , .scala , Build Plugin .scala )

jar lib/( ) , build.sbt :

libraryDependencies += "org.apache.derby" % "derby" % "10.4.1.3"

10.4.1.3 Apache Derby

key libraryDependencies := , % += key ,
% Ivy ID ,
,
```

Scope

scope .sbt

Key

```
name key sbt map ,
, key , "scope"
:
```

- , key
- ,key compile main source test source
- Key packageOptions(jar) , class packageBin, packageSrc


```

    key name , scope
, scoped key
    ,sbt map settings , map key scoped key set-
ting( build.sbt ) scoped key
    scope , , build.sbt scope

```

Scope

```

Scope , scope( , key )
    scope :

```

- Projects
- Configurations
- Tasks

```

Project Scope , settings ,keys scope
Project , setting setting , setting

```

```

Configuration Scope configuration , classpath,sources,
packages Configuration Ivy MavenScopes
sbt configurations:

```

- Compile (src/main/scala)
- Test (src/test/scala)
- Runtime task run classpath

```

, , , key configuration, configuration task
key:compile,package run; key key( sourceDirectories
scalacOptions fullClasspath) configuration

```

```

Task Scope Settings task ,task packageSrc setting
packageOptions
    , task key( packageSrc) key( packageOptions) scope
    task(packageSrc,packageBin,packageDoc) key, artifactName
packageOptions key task

```

Scope

```

scope ( task task ), Global
Global : setting task Global, setting task

scope key , key
scope,sbt scope , key scope ,sbt scope( Global
scope scope)
scope setting , scope
inspect key “ ”

```

sbt scoped key

```
,sbt ( )scoped keys:
```

```
{<build-uri>}<project-id>/config:intask::key
```

- {<build-uri>}/<project-id> project project scope,
- <project-id>
- config configuration
- intask task
- key scoped key

```
* , Global scope
scoped key, :
```

- project, project
- configuration task, key configuration

```
, Configuration
```

scoped key

- fullClasspath key, scope: project,key configuration task scope
- test:fullClasspath configuration, fullClasspath test configuration scope , scope

- *:fullClasspath configuration Global, configuration
- doc::fullClasspath key fullClasspath doc task ,project configuration
- {file:/home/hp/checkout/hello/}default-aea33a/test:fullClasspath project, {file:/home/hp/checkout/hello/}default-aea33a ,{file:/home/hp/checkout/hello/} project, project id default-aea33a configuration test, task
- {file:/home/hp/checkout/hello/}/test:fullClasspath {file:/home/hp/checkout/hello/} project
- {./}/test:fullClasspath {./} project {./} Scala ThisBuild
- {file:/home/hp/checkout/hello/}/compile:doc::fullClasspath scope

scope

sbt , inspect key scope inspect test:fullClasspath:

```
$ sbt
> inspect test:fullClasspath
[info] Task: scala.collection.Seq[sbt.Attributed[java.io.File]]
[info] Description:
[info] The exported classpath, consisting of build products and unmanaged and managed, internal
[info] Provided by:
[info] {file:/home/hp/checkout/hello/}default-aea33a/test:fullClasspath
[info] Dependencies:
[info] test:exportedProducts
[info] test:dependencyClasspath
[info] Reverse dependencies:
[info] test:runMain
[info] test:run
[info] test:testLoader
[info] test:console
[info] Delegates:
[info] test:fullClasspath
[info] runtime:fullClasspath
[info] compile:fullClasspath
[info] *:fullClasspath
[info] {./}/test:fullClasspath
[info] {./}/runtime:fullClasspath
[info] {./}/compile:fullClasspath
[info] {./}/*:fullClasspath
[info] */test:fullClasspath
[info] */runtime:fullClasspath
```

```

[info] */compile:fullClasspath
[info] */*:fullClasspath
[info] Related:
[info] compile:fullClasspath
[info] compile:fullClasspath(for doc)
[info] test:fullClasspath(for doc)
[info] runtime:fullClasspath

```

```

, task( .sbt setting ) task scala.collection.Seq[sbt.Attributed[java.io.File]]
“Provided by” scoped key, {file:/home/hp/checkout/hello/}default-aea33a/test:fullClasspath
test configuration {file:/home/hp/checkout/hello/}default-aea33a
project )

```

```

“Dependencies” ;

```

```

; ,sbt :

```

- configuration(runtime:fullClasspath compile:fullClasspath)
 - scoped key ,project “ project” task Global
- project “ project” task Global ,configuration Global(*:fullClasspath)
- project ,project {..} ThisBuild
- project Global(*:test:fullClasspath)(, project cur-
 - rent, Global ; :* “ project” project ; :*/test:fullClasspath
 - test:fullClasspath)
- project configuration Global (*:fullClasspath)(task
 - Global, */*:fullClasspath Global)

```

inspect fullClasspath( inspect test:fullClasspath ) con-
figuration ,sbt compile inspect compile:fullClasspath
inspect fullClasspath
inspect *:fullClasspath ,fullClasspath Global configuration

```

Configuration

scope

```

build.sbt key, project ,configuration task Global:

```

```

name := "hello"

```

```

sbt inspect name {file:/home/hp/checkout/hello/}default-aea33a/*:name
, ,project {file:/home/hp/checkout/hello/}default-aea33a, configu-
ration *( ),task ( )

```

```

build.sbt      , " " build.sbt      ( , build.sbt )
Keys      in scope in      scope      , name Compile configuration
,      :

name in Compile := "hello"

      name      packageBin task ( ! ):

name in packageBin := "hello"

      name      scope , Compile configuration      packageBin task :

name in (Compile, packageBin) := "hello"

      Global      :

name in Global := "hello"

(name in Global      scope      Global      scope      Global;task
configuration      Global,      project      Global,      ,      */*:name
{file:/home/hp/checkout/hello/}default-aea33a/*:name)

      Scala, : in := , Scala      , Java :

name.in(Compile).:=("hello")

,

scope

key      , scope ,compile task      Compile Test configuration scope
, scope

key compile , compile in Compile compile in Test compile
project scope task, configuration scope compile task

" " , scope , scope key scope sbt
; " compile:compile?"

,name key , key name scope (scope ) ,packageOptions
in (Compile, packageBin) key name ,packageOptions key
name, ( in key, scope: project,global config,global task)

```

```

:= , .sbt scope

:

, Setting , Setting sbt ( map) Setting sbt
map map map sbt
setting map , :=
:= Setting map , name := "hello" map , map
key name "hello"
Settings (build.sbt , .scala Setting sbt
)

: += +=

:= , key SettingKey[T] T , , key se-
quence,

• +=
• +=

, key sourceDirectories in Compile Seq[File] key
src/main/scala source ( ), :

sourceDirectories in Compile += new File("source")

, sbt file() :

sourceDirectories in Compile += file("source")

(file() File)
+= :

sourceDirectories in Compile += Seq(file("sources1"), file("sources2"))

Seq(a, b, c, ...) Scala
source , :=:

sourceDirectories in Compile := Seq(file("sources1"), file("sources2"))

```

```

    key

    task setting value value :=,+= +=
    , project organization

// name our organization after our project (both are SettingKey[String])
organization := name.value

    , :

// name is a Key[String], baseDirectory is a Key[File]
// name the project after the directory it's inside
name := baseDirectory.value.getName

    java.io.File getName baseDirectory
    ,

name := "project " + name.value + " from " + organization.value + " version " + version.value

    name organization version , name

    name := baseDirectory.value.getName ,name baseDirectory
build.sbt , sbt , inspect name, ( ):

[info] Dependencies:
[info] *:baseDirectory

    sbt setting setting setting task, task
    , inspect compile key compileInputs, inspect compileInputs
    key compile , sbt update compile sbt
    update
    ,sbt key , key !

    :=,+= += key , ,sbt , “ ”
    , key scope
    , ;sbt ,

```

```

    key task task setting task task Def.task
taskValue , :=, += +=

    , classpath source generator

sourceGenerators in Compile += Def.task {
  myGenerator(baseDirectory.value, (managedClasspath in Compile).value)
}.taskValue

    task .sbt , := task key Setting[Task[T]]
Setting[T] Task setting , setting task

    key( Keys ):

val scalacOptions = taskKey[Seq[String]]("Options for the Scala compiler.")
val checksums = settingKey[Seq[String]]("The list of checksums to generate and to verify for

(scalacOptions checksums , key, task )

    build.sbt scalacOptions checksums, , :

// scalacOptions task checksums setting
scalacOptions := checksums.value

    , setting key task key setting key , task
    , task

// checksums setting scalacOptions task
checksums := scalacOptions.value

:= +=

    setting task key, :=
    , , :

cleanFiles += file("coverage-report-" + name.value + ".txt")

    , , .sbt ,Scopes

    :

• lib jar
• , (repository)

```



```

        : jar lib , classpath !
        jar lib , ScalaCheck,Specs2,ScalaTest

lib classpaths( compile, test, run console) classpath,
, dependencyClasspath in Compile dependencyClasspath in
Runtime
, build.sbt , unmanagedBase key, lib
custom_lib lib:

unmanagedBase := baseDirectory.value / "custom_lib"

baseDirectory , baseDirectory unmanagedBase,
value

unmanagedBase jar task unmanagedJars ,
task unmanagedJars task, Compile configuration , lib :

unmanagedJars in Compile := Seq.empty[sbt.Attributed[java.io.File]]

sbt Apache Ivy , Ivy Maven ,

libraryDependencies Key , libraryDependencies
Maven POM Ivy , sbt
,groupId,artifactId revision :

libraryDependencies += groupId % artifactID % revision

, Configuration val configuration:

libraryDependencies += groupId % artifactID % revision % configuration

libraryDependencies Keys :

val libraryDependencies = settingKey[Seq[ModuleID]]("Declares managed dependencies.")

```

```

%      ModuleID , ModuleID libraryDependencies
, sbt( Ivy)          sbt          , ,Apache Derby   Maven2   :

libraryDependencies += "org.apache.derby" % "derby" % "10.4.1.3"

    build.sbt      , update,sbt Derby   ~/.ivy2/cache/org.apache.derby/ ( ,
compile update,          update )
,      +=      :

libraryDependencies += Seq(
    groupId % artifactID % revision,
    groupId % otherID % otherRevision
)

,      libraryDependencies :=

%%      Scala      groupId %% artifactID % revision   groupId %
artifactID % revision(   groupId   %%),sbt      Scala
%%:

libraryDependencies += "org.scala-tools" % "scala-stm_2.11.1" % "0.3"

    scalaVersion 2.11.1,      ( "org.scala-tools"   %%):

libraryDependencies += "org.scala-tools" %% "scala-stm" % "0.3"

    Scala ,

    Scala      ; %%      2.10.1,      scalaVersion :=
"2.10.4",      %% 2.10.1      %% ,      Scala ,      ( )

Ivy      groupId % artifactID % revision      revision      Ivy
      "latest.integration","2.9.+"      "[1.0,)",
"1.6.1" Ivy

```

```

        , sbt      Maven2      ,      resolver      Ivy
      :

    resolvers += name at location

        at

      :

    resolvers += "Sonatype OSS Snapshots" at "https://oss.sonatype.org/content/repositories/snapshots"

    resolvers key Keys      :

    val resolvers = settingKey[Seq[Resolver]]("      ")

    at      Resolver

    sbt      Maven      :

    resolvers += "Local Maven Repository" at "file://" + Path.userHome.absolutePath + "/.m2/repository"

    ,      :

    resolvers += Resolver.mavenLocal

    resolvers      ,

    sbt resolvers      externalResolvers

    ,      ,      externalResolvers      resolvers

Per-configuration dependencies      ( src/test/scala , Test configuration )

    Test configuration classpath      Compile configuration,      % "test":

    libraryDependencies += "org.apache.derby" % "derby" % "10.4.1.3" % "test"

    Test configuration:

    libraryDependencies += "org.apache.derby" % "derby" % "10.4.1.3" % Test

    ,      sbt      show compile:dependencyClasspath,      derby jar      show
    test:dependencyClasspath,      derby jar

    ,      , ScalaCheck, Specs2 ScalaTest      % "test"

```

```

    ,      .sbt

    ,      ,
    ,      jar ,
    Project lazy val      , :

lazy val util = project
lazy val core = project

val      ID      ID      in      ,      :

lazy val util = project.in(file("util"))
lazy val core = project in file("core")

    ,      :aggregate classpath

Aggregation Aggregation      aggregate      task      aggregated      ,

lazy val root = (project in file(".")).aggregate(util, core)
lazy val util = project
lazy val core = project

    ,root      util      core      ,      sbt,
    ,      root      ,      task      ,      update task:

lazy val root = (project in file(".")).
    aggregate(util, core).
    settings(
        aggregate in update := false
    )

[...]

aggregate in update      update task      scope      key (      scopes )
:      task,task

```

```

Classpath dependsOn , core classpath until,
core:

```

```

lazy val core = project.dependsOn(util)

```

```

core util ; core ,util
, dependsOn(bar, baz) dependsOn

```

```

configuration classpath foo dependsOn(bar) foo compile
configuration bar compile configuration :dependsOn(bar %
"compile->compile")
"compile->compile" -> "depends on", "test->compile" foo test
configuration bar compile configuration
->config ->compile, dependsOn(bar % "test") foo test configu-
ration bar Compile configuration
"test->test" test test , bar/src/test/scala ,
foo/src/test/scala ,
configuration, , :dependsOn(bar % "test->test;compile->compile")

```

```

root

```

```

,sbt hello-foo base = file("foo"), foo
foo , foo/Foo.scala, foo/src/main/scala sbt foo

foo .sbt , foo/build.sbt, , hello-foo scope
hello , hello/build.sbt,hello/bar/build.sbt hello/foo/build.sbt
(version := "0.6") sbt show version ( ):

```

```

> show version
[info] hello-foo/*:version
[info] 0.7
[info] hello-bar/*:version
[info] 0.9
[info] hello/*:version
[info] 0.5

```

```

hello-foo/*:version hello/foo/build.sbt ,hello-bar/*:version
hello/bar/build.sbt ,hello/*:version hello/build.sbt scoped
keys version key scope , build.sbt build.sbt
.sbt , .scala , .scala

```

```

,      ,      .scala
,      project/*.scala  foo/project/Build.scala

sbt      ,  projects      ,  project  <projectname>      task
compile,      root  ,
      ID      task,  subProjectID/compile

.sbt      .sbt      .sbt      ,      project/      Scala      sbt
,
:
<root>/project/Common.scala:

import sbt._
import Keys._

object Common {
  def text = "org.example"
}

<root>/build.sbt:

organization := Common.text

.scala

,      build.sbt

,      task  ,      codeCoverage task,

```

```

    hello ,      sbt-site , hello/project/site.sbt      Ivy ID
    addSbtPlugin:

addSbtPlugin("com.typesafe.sbt" % "sbt-site" % "0.7.0")

    sbt-assembly,      hello/project/assembly.sbt :

addSbtPlugin("com.eed3si9n" % "sbt-assembly" % "0.11.2")

    ,      :

resolvers += Resolver.sonatypeRepo("public")

    ,

0.13.5    sbt,      ,

    ,      build.sbt :

lazy val util = (project in file("util")).
  enablePlugins(FooPlugin, BarPlugin).
  settings(
    name := "hello-util"
  )

enablePlugins

    disablePlugins      ,      util    IvyPlugin      , build.sbt :

lazy val util = (project in file("util")).
  enablePlugins(FooPlugin, BarPlugin).
  disablePlugins(plugins.IvyPlugin).
  settings(
    name := "hello-util"
  )

    ,    sbt    plugins

:

```

```

> plugins
In file:/home/jsuereth/projects/sbt/test-ivy-issues/
  sbt.plugins.IvyPlugin: enabled in scala-sbt-org
  sbt.plugins.JvmPlugin: enabled in scala-sbt-org
  sbt.plugins.CorePlugin: enabled in scala-sbt-org
  sbt.plugins.JUnitXmlReportPlugin: enabled in scala-sbt-org

, plugins      sbt      sbt      3      :

  1. CorePlugin:      task
  2. IvyPlugin:
  3. JvmPlugin:      Java/Scala

, JUnitXmlReportPlugin      junit-xml

      ,      ,      ,

, sbt-site ,      ,      site.sbt

site.settings

      ,      :

// `util`      site
lazy val util = (project in file("util"))

// `core`      site
lazy val core = (project in file("core")).
  settings(site.settings : _*)

      ,      ~/.sbt/0.13/plugins/      ~/.sbt/0.13/plugins/      classpath
sbt      , ~/.sbt/0.13/plugins/      .sbt      .scala      project/

      ,      ~/.sbt/0.13/plugins//build.sbt      addSbtPlugin()

      ,

:

```


- IDE (sbt IDE)
- web , [xsbt-web-plugin](#)

, , ,

, sbt , .sbt

SettingKey TaskKey .sbt InputKey
Keys :

```
val scalaVersion = settingKey [String]("scala ")
val clean = taskKey[Unit](" , source , ")

: ( "scalaVersion" ) ( " scala " )
.sbt , T SettingKey[T] T TaskKey [T] .sbt
, , " "( )
.sbt ,.scala .scala val,Build Plugin plugin val
.sbt

, , ; := :

val sampleStringTask = taskKey[String]("A sample string task.")
val sampleIntTask = taskKey[Int]("A sample int task.")

sampleStringTask := System.getProperty("user.home")

sampleIntTask := {
  val sum = 1 + 2
  println("sum: " + sum)
  sum
}
```

```

    , value
    sbt ; Scala , HTML, HTML
( , HTML )
sbt , API IO

```

```

, [ ][sing-Plugins]
;

```

.scala

```

, .sbt

```

sbt

```

build.sbt , sbt sbt Scala .sbt ?
project project sbt
, , project/project/
:

```

```

hello/ #
Hello.scala #
build.sbt # build.sbt project/
project/ #
Build.scala #
build.sbt #
project/ #
Build.scala # project/project/
! project/project/
.scala .sbt , build.sbt Build.scala , ,

```

```

.scala

.sbt      :

hello/          #

    build.sbt      # build.sbt project/

    project/      #

        Build.scala    #

build.sbt  Scala      , Build.scala    (  project/  .scala  )
    *.sbt

.sbt      ,      ,

build.sbt  Build.scala

.sbt  .scala      ,
    ,      hello      , hello/project/Build.scala:

import sbt._
import Keys._

object HelloBuild extends Build {
    val sampleKeyA = settingKey[String]("demo key A")
    val sampleKeyB = settingKey[String]("demo key B")
    val sampleKeyC = settingKey[String]("demo key C")
    val sampleKeyD = settingKey[String]("demo key D")

    override lazy val settings = super.settings ++
        Seq(
            sampleKeyA := "A: in Build.settings in Build.scala",
            resolvers := Seq()
        )

    lazy val root = Project(id = "hello",
        base = file("."),
        settings = Seq(
            sampleKeyB := "B: in the root project settings in Build.scala"
        ))
}

```

```

, hello/build.sbt:

sampleKeyC in ThisBuild := "C: in build.sbt scoped to ThisBuild"

sampleKeyD := "D: in build.sbt"

sbt      inspect sampleKeyA,  :

[info] Setting: java.lang.String = A: in Build.settings in Build.scala
[info] Provided by:
[info] {file:/home/hp/checkout/hello/}/*:sampleKeyA

      inspect sampleKeyC,  :

[info] Setting: java.lang.String = C: in build.sbt scoped to ThisBuild
[info] Provided by:
[info] {file:/home/hp/checkout/hello/}/*:sampleKeyC

      "Provided by"      value      .sbt      sampleKeyC in ThisBuild
Build.settings , .scala  sbt          ,

,inspect sampleKeyB:

[info] Setting: java.lang.String = B: in the root project settings in Build.scala
[info] Provided by:
[info] {file:/home/hp/checkout/hello/}hello/*:sampleKeyB

      sampleKeyB      : ({file:/home/hp/checkout/hello/}hello)
({file:/home/hp/checkout/hello/})

      ,inspect sampleKeyD  sampleKeyB:

[info] Setting: java.lang.String = D: in build.sbt
[info] Provided by:
[info] {file:/home/hp/checkout/hello/}hello/*:sampleKeyD

sbt      .sbt      Build.settings  Project.setting      ,.sbt
Build.scala,      sanokeC  sampleD,      build.sbt      build.sbt
Build.sbt

      :sampleKeyC sampleKeyD      build.sbt      sbt      Build      .sbt
      ,import HelloBuild._      build.sbt

:- .scala , Build.settings ,      - .scala , Project.settings
,      - .scala Build      .sbt - .sbt      .scala -
.sbt      ,

```

```

    .scala

    .scala ,      Scala ,      ,      ,      .scala      .sbt , .scala
    ,      .sbt

    sbt      , project/      reload plugins

> reload plugins
[info] Set current project to default-a0e8e4 (in build file:/home/hp/checkout/hello/project/)
> show sources
[info] ArrayBuffer(/home/hp/checkout/hello/project/Build.scala)
> reload return
[info] Loading project definition from /home/hp/checkout/hello/project
[info] Set current project to hello (in build file:/home/hp/checkout/hello/)
> show sources
[info] ArrayBuffer(/home/hp/checkout/hello/hw.scala)
>

    reload return      ,

:

    build.sbt      Build Project      settings      ,Build Project
    settings      , build.sbt      ,      sbt      Build Project
    “      ”

    ,      :

    • Build.settings Project.settings      .scala
    •      ; ~/.sbt/0.13/global.sbt
    •      ,
    • .sbt
    • ( project ) ( ~/.sbt/0.13/plugins/)

    sbt,      ,      ,      sbt      sbt
    ,

```

sbt:

- Scala, Scala [Programming in Scala](#), Scala
- [.sbt](#)
- Setting, sbt Setting task
- Setting, key ::=, += ++=
- , ; , Setting sbt
- , key
- tasks, key value task Non-task
- [Scopes](#)
- key value, scope
- scope :configuration, project, task
- scope task configuration
- configuration, Compile Test
- project “ ” scope
- scopes scope
- [.sbt](#) vs. [.scala](#)
- build.sbt, .scala task
- sbt,
-
- addSbtPlugin project/plugins.sbt (build.sbt)
- , , sbt
- !

sbt, !