



CEGŁADANYCH  
DANE W CHMURZE

# Apache Spark na Windowsie czy to możliwe?

23.09.2021 Krzysztof Nojman



Witam, do tej pory pisałem o Databricks jako o super narzędziu do Big Data. Jest on niewątpliwie bardzo użyteczny, ale do tego potrzeba przeglądarki i dostępu do chmury publicznej, Azure, AWS lub GCP. A co jeśli chcesz zacząć przygodę ze Apache Spark na Windowsie bez wydawania kasy na chmurę. Mam dla Ciebie dobre wieści jest to możliwe, żeby używać Big Data na kilku rdzeniach. **Spark on Windows** ?? tak to możliwe. Dzięki genialnemu rozwiązaniu, jaki ma Spark da się go używać w opcji 'Local Mode' właśnie po to by można było go uruchomić lokalnie. W takim modelu działa [Databricks Community](#).

Sparka można używać w konsoli co może się wydawać ułatwieniem, lecz ja bym proponował od razu zacząć w porządnym narzędziu. IntelliJ lub PyCharm. Dzięki temu

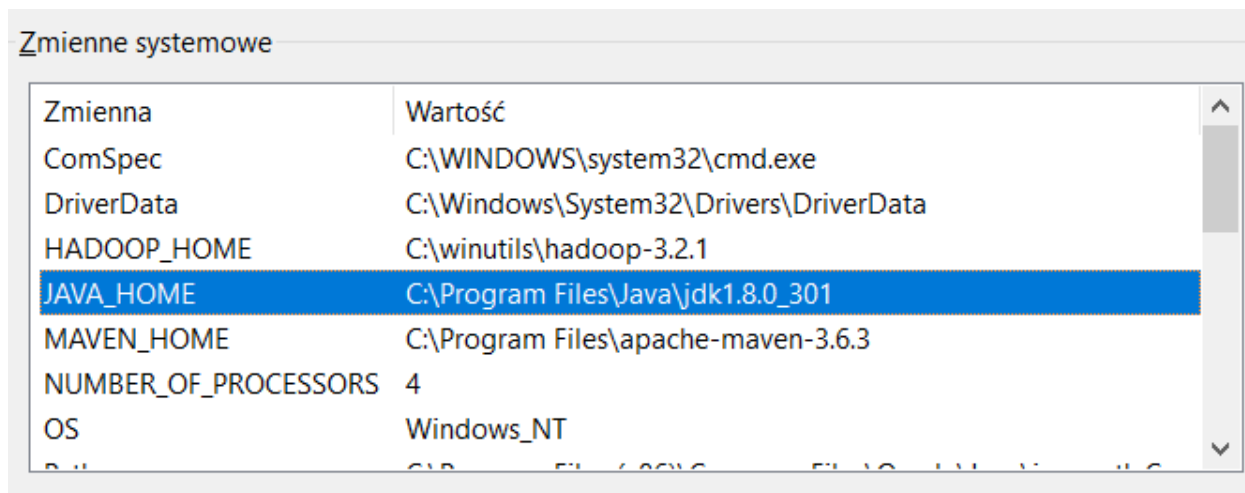
# Wymagania

Oto lista narzędzi, którą trzeba zainstalować **Java**, **Python**, **Spark**, **winutils**, w zależności, w jakim języku chcesz pracować, dla Scali proponuję **IntelliJ** dla Pythona **PyCharm**. Z dodatkowych narzędzi do budowania mam zainstalowane sbt i maven to oczywiście jest świat IntelliJ. **Sbt** współpracuje lepiej ze scalą, więc na początek polecam prace z sbt. Jeśli chodzi o maven, to pracując w Azure ma on natywne wsparcie w pipelinech devopsowych (Artifacts), więc w Azure będzie lepszym wyborem.

## Java

Jest bardzo małe prawdopodobieństwo że nie masz Java, jeśli tak jest to instrukcje znajdziesz [tutaj](#). Wymagana wersja 1.8 lub 11. Po zainstalowaniu trzeba przejść kolejne kroki.

1. Upewnij się, że Java jest dodana do Zmiennych środowiskowych ('Environment Variables').
  - Dodaj JAVA\_HOME do Zmienne Systemowe



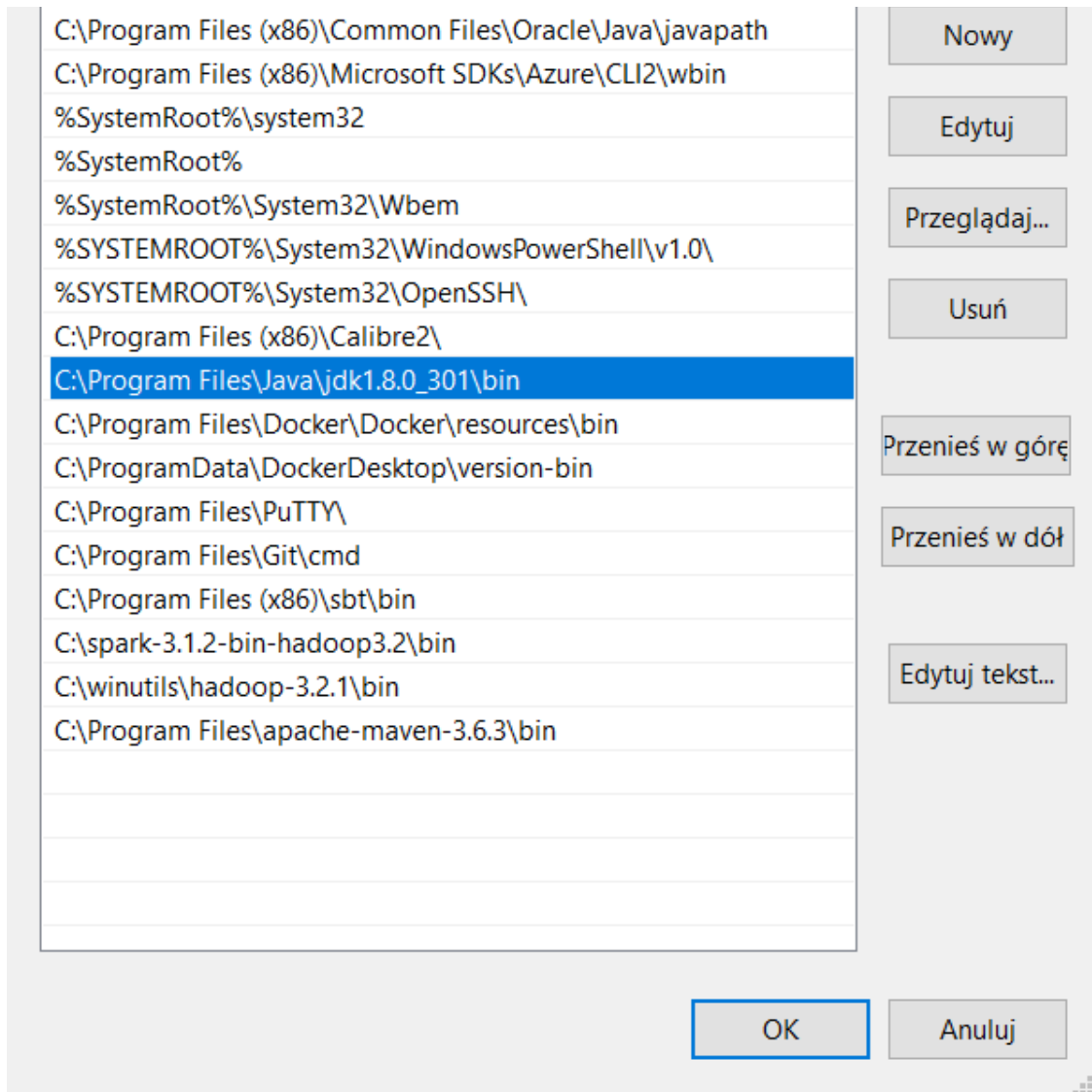
2. Dodaj ścieżkę folderu bin do **Path**

START

BLOG

KIM JESTEM

KONTAKT



3. Sprawdź w konsoli czy Java jest zainstalowana. Konsole możesz uruchomić wpisując w wyszukiwarkę Windows **'cmd'** potem wpisz **'java -version'**. Jeśli wszystko jest dobrze będzie to wyglądało tak:

```
C:\Users\admin>java -version
java version "1.8.0_301"
Java(TM) SE Runtime Environment (build 1.8.0_301-b09)
Java HotSpot(TM) 64-Bit Server VM (build 25.301-b09, mixed mode)

C:\Users\admin>
```

## Apache Spark

1. Pobierz najnowszego Sparka ze strony <https://spark.apache.org/downloads.html>

START

BLOG

KIM JESTEM

KONTAKT



Download

Libraries ▾

Documentation ▾

Examples

Community ▾

Developers ▾

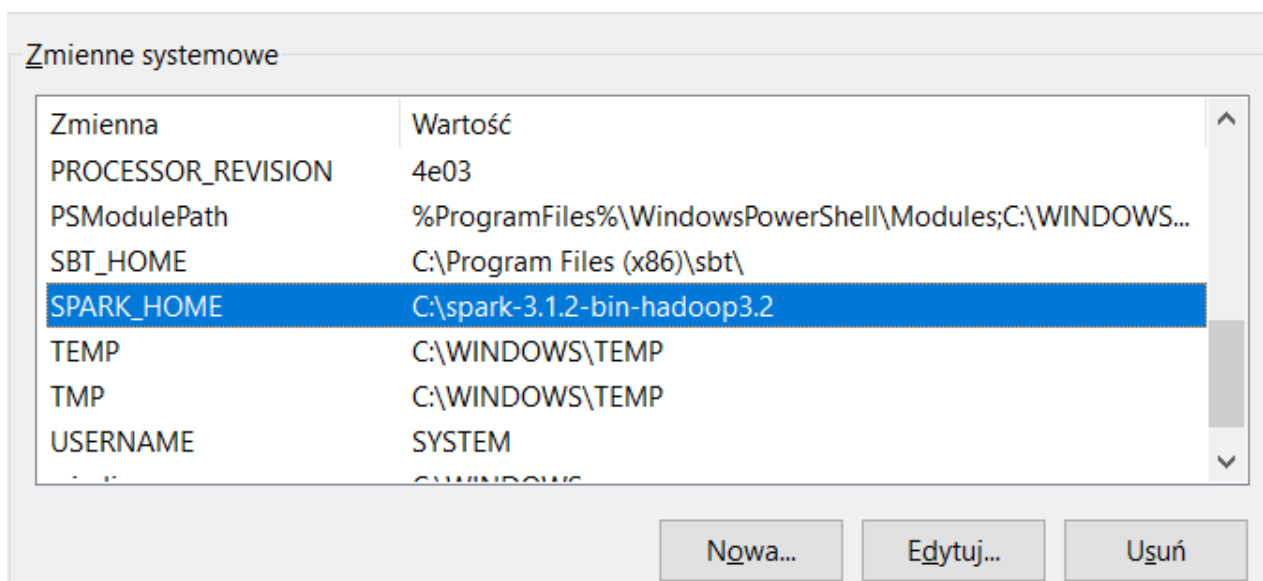
## Download Apache Spark™

1. Choose a Spark release:
2. Choose a package type:
3. Download Spark: [spark-3.0.3-bin-hadoop3.2.tgz](#)
4. Verify this release using the 3.0.3 [signatures](#), [checksums](#) and [project release KEYS](#).

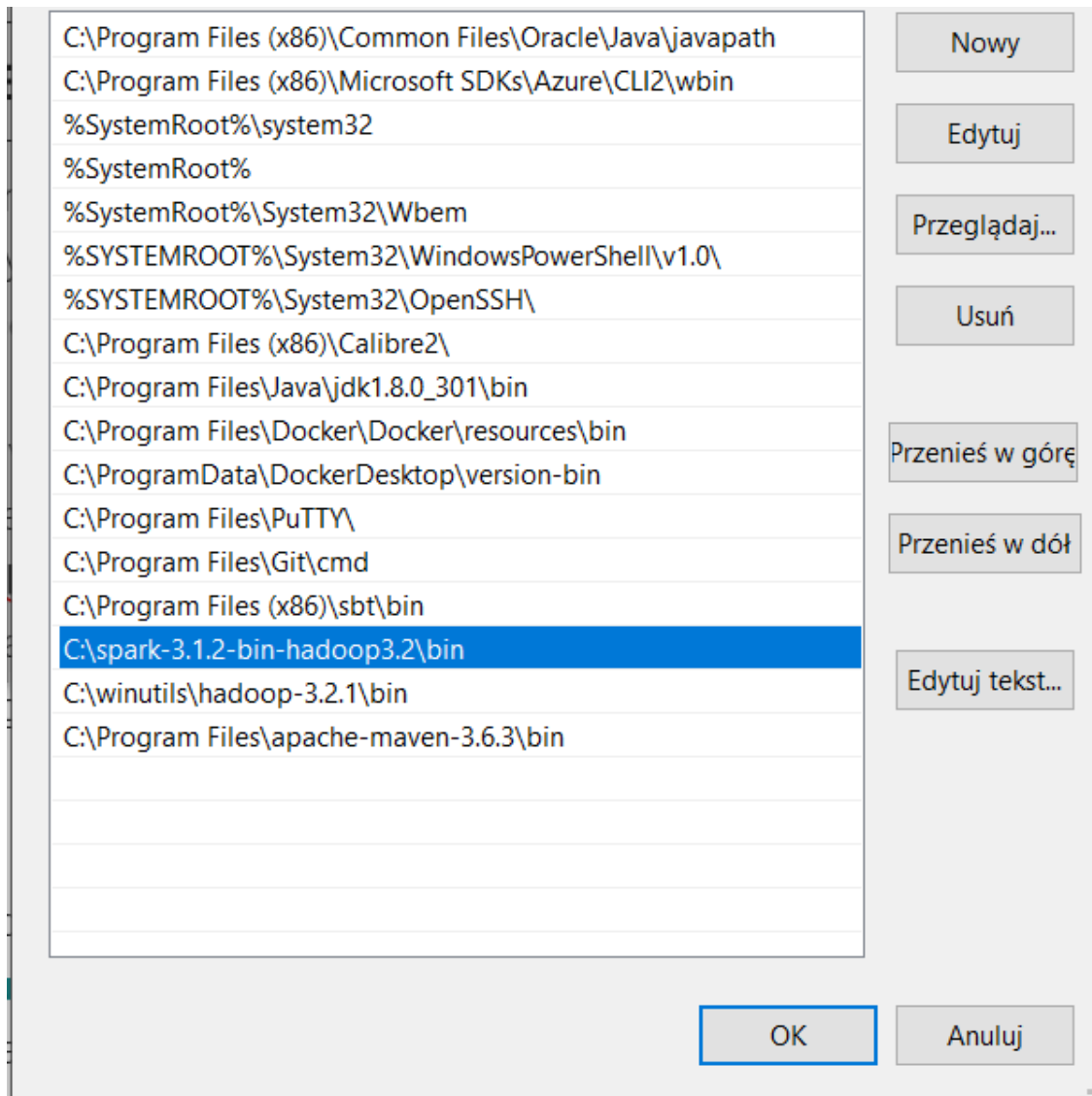
Note that, Spark 2.x is pre-built with Scala 2.11 except version 2.4.2, which is pre-built with Scala 2.12. Spark 3.0+ is pre-built with Scala 2.12.

### Spark

1. Rozpakuj pobrany plik, w moim przypadku jest to **spark-3.1.2-bin-hadoop3.2.tgz**
2. Umieść wszystkie foldery w ścieżce na dysku, ja trzymam to na C:\spark-3.1.2-bin-hadoop3.2, ale możesz to umieścić w innym miejscu.
3. Dodaj Zmienną środowiskową ('Environment Variables').
  - Dodaj SPARK\_HOME do Zmienne Systemowe



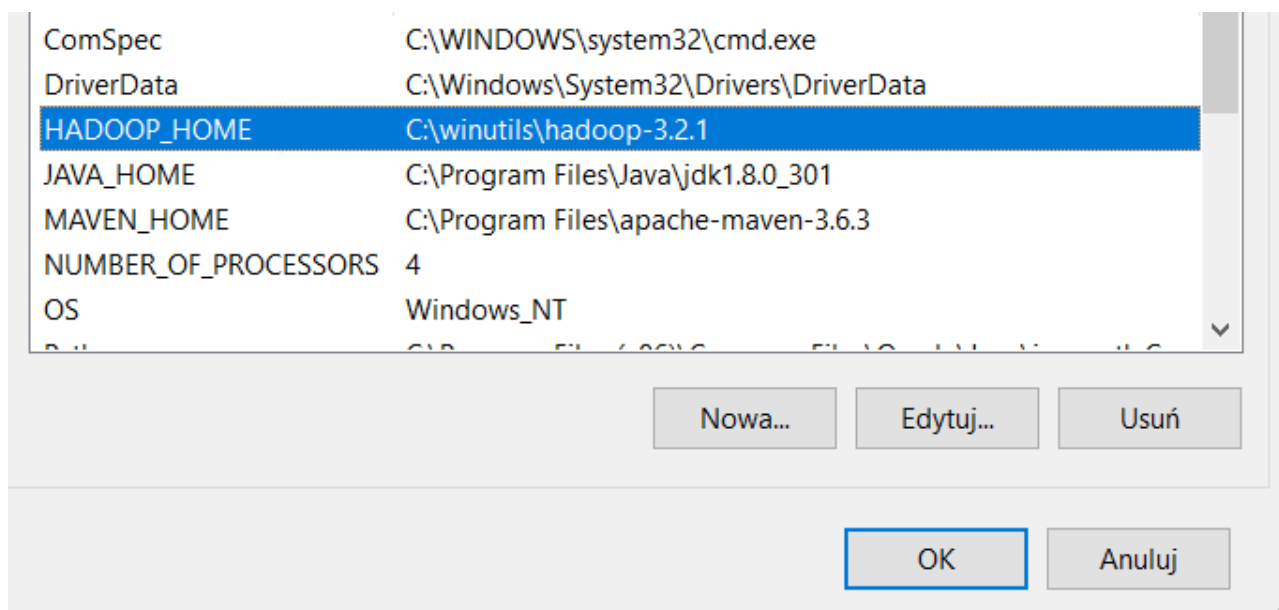
4. Dodaj ścieżkę folderu bin do **Path**



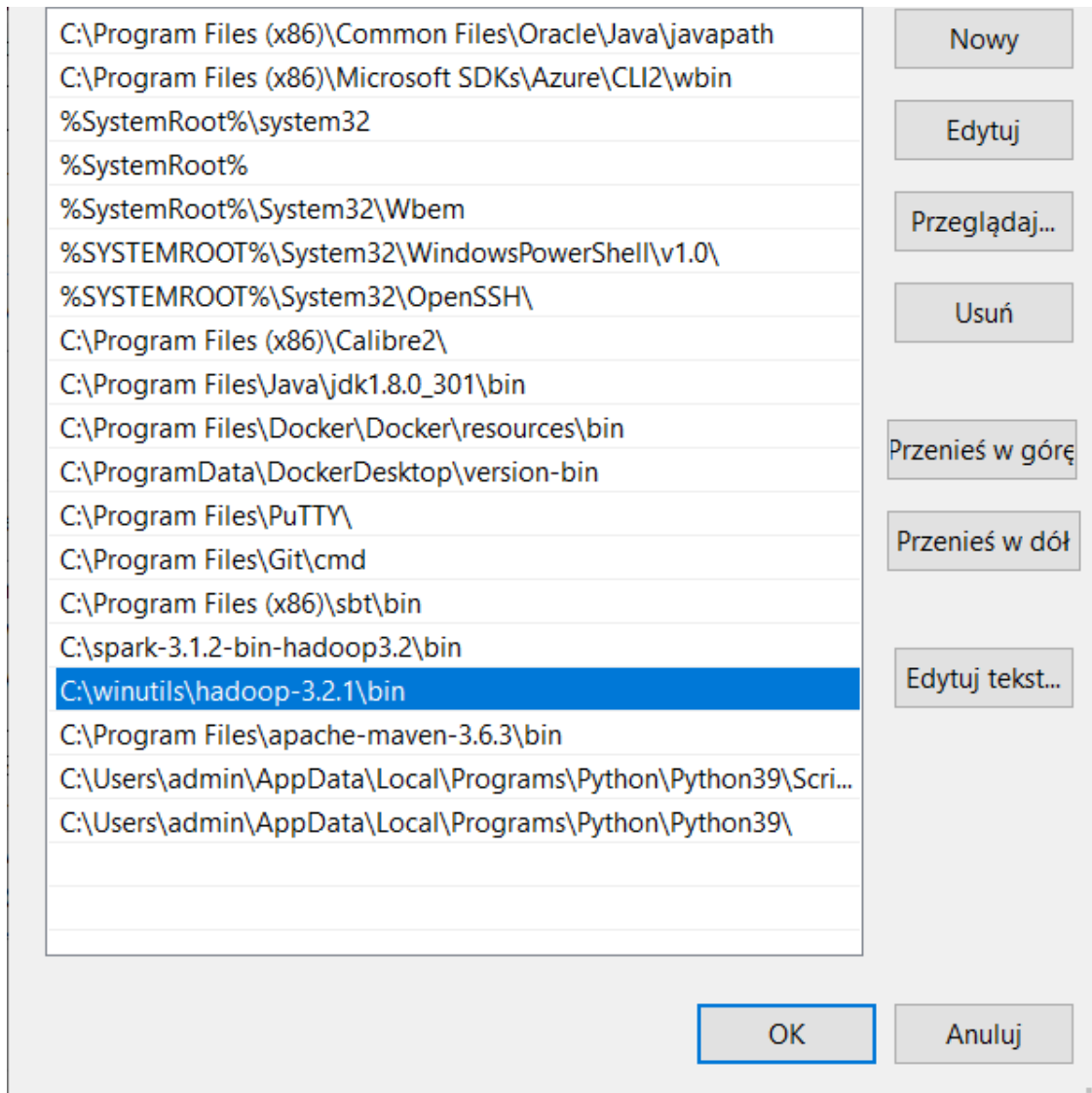
## Winutils

Winutils to plik binarny, który umożliwia działanie kodu hadoop (HDFS) na windowsie. Możesz pobrać gotowy plik ja używam tego: [winutils](#), lub się pobawić i wygenerować własny. Jest sporo instrukcji na necie ja trafiłem na tą [stronę](#) ale tego nie testowałem.

1. Pobierz pliki lub stwórz własnoręcznie w zależności od preferencji. Jak już zdobędziesz plik to umieść go na dysku, ja wrzucam na C:/
2. Dodaj Zmienną środowiskową ('Environment Variables').
  - Dodaj HADOOP\_HOME do Zmienne Systemowe

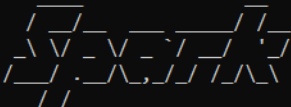
[START](#)[BLOG](#)[KIM JESTEM](#)[KONTAKT](#)

- Dodaj ścieżkę folderu bin do **Path**



3. Sprawdź w konsoli czy Spark działa. Konsolę uruchomisz jak w przypadku powyżej z Java. Wpisz 'spark-shell'. Jeśli wszystko jest dobrze będzie to wyglądało tak:

```
C:\Wiersz polecenia - spark-shell  
Microsoft Windows [Version 10.0.19042.1165]  
(c) Microsoft Corporation. Wszelkie prawa zastrzeżone.  
  
C:\Users\admin>spark-shell  
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties  
Setting default log level to "WARN".  
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).  
Spark context Web UI available at http://host.docker.internal:4040  
Spark context available as 'sc' (master = local[*], app id = local-1631778405032).  
Spark session available as 'spark'.  
Welcome to
```



version 3.1.2

```
Using Scala version 2.12.10 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_301)  
Type in expressions to have them evaluated.  
Type :help for more information.
```

```
scala> 21/09/16 09:46:56 WARN ProcfsMetricsGetter: Exception when trying to compute pagesize, as a result reporting of P  
rocessTree metrics is stopped
```



może się wydawać ekscentryczna. Ja bym polecił jakieś porządne narzędzie. Jeśli zainstalujesz Pythona, to w konsoli będziesz mógł uruchomić **Pyspark**.

## Python

Bez Pythona świat nie byłby taki sam, więc warto go poznać. Obecnie, Python jest bardzo popularnym językiem używanym w środowisku Spark, więc jeśli jesteś nim zainteresowany to polecam go wypróbować. Pobierz Pythona z oficjalnej [strony](#).

1. Podczas instalacji wybierz opcje dodaj Python to PATH.
2. Sprawdź w konsoli wersje Pythona. Jeśli wszystko poszło dobrze zobaczysz version w konsoli.

### Wiersz polecenia

```
Microsoft Windows [Version 10.0.19042.1165]
(c) Microsoft Corporation. Wszelkie prawa zastrzeżone.

C:\Users\admin>python --version
Python 3.9.7

C:\Users\admin>
```

Teraz powinieneś uruchomić Pyspark żeby sprawdzić czy wszystko działa. Wróć do konsoli i wpisz **pyspark**. Zobaczysz środowisko w którym możesz używać Python w Sparku.



```
C:\Users\admin>pyspark
Python 3.9.7 (tags/v3.9.7:1016ef3, Aug 30 2021, 20:19:38) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Welcome to

  ____      _
 / ___|    / \
| |  | |  / _ \
| |  | | / ___\
| |  | || |___|
| |  | || |___|
 \___|_||_|

version 3.1.2

Using Python version 3.9.7 (tags/v3.9.7:1016ef3, Aug 30 2021 20:19:38)
Spark context Web UI available at http://host.docker.internal:4040
Spark context available as 'sc' (master = local[*], app id = local-1631779595416).
SparkSession available as 'spark'.
>>>
```

## IntelliJ

Jeśli chcesz działać w świecie scali to zapraszam do IntelliJ, jest to bardzo popularne narzędzie i działa jak należy.

Wystarczy pobrać **IntelliJ Community** i zainstalować. Tutaj nie ma specjalnych wymagań, wybierz wszystkie opcje i będzie dobrze.



Version: 2021.2.1  
Build: 212.5080.55  
24 August 2021  
[Release notes](#)

### Download IntelliJ IDEA

[Windows](#)[macOS](#)[Linux](#)

#### Ultimate

For web and enterprise development

[Download](#)[.exe](#)

Free 30-day trial

#### Community

For JVM and Android development

[Download](#)[.exe](#)

Free, built on open source

Po zainstalowaniu najważniejsze elementy, które trzeba doinstalować to plugin Scala. Żeby zainstalować plugin:

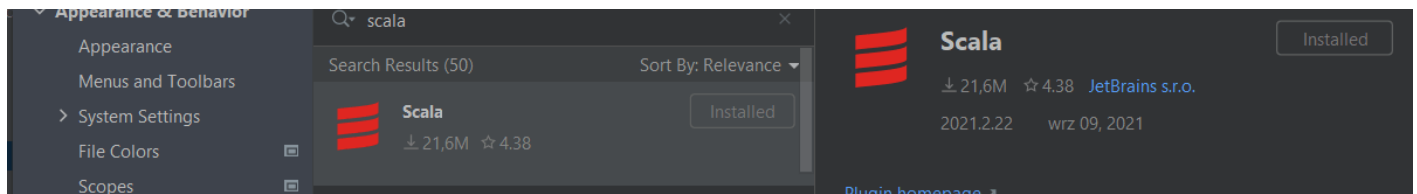
- File > Settings > Plugins

START

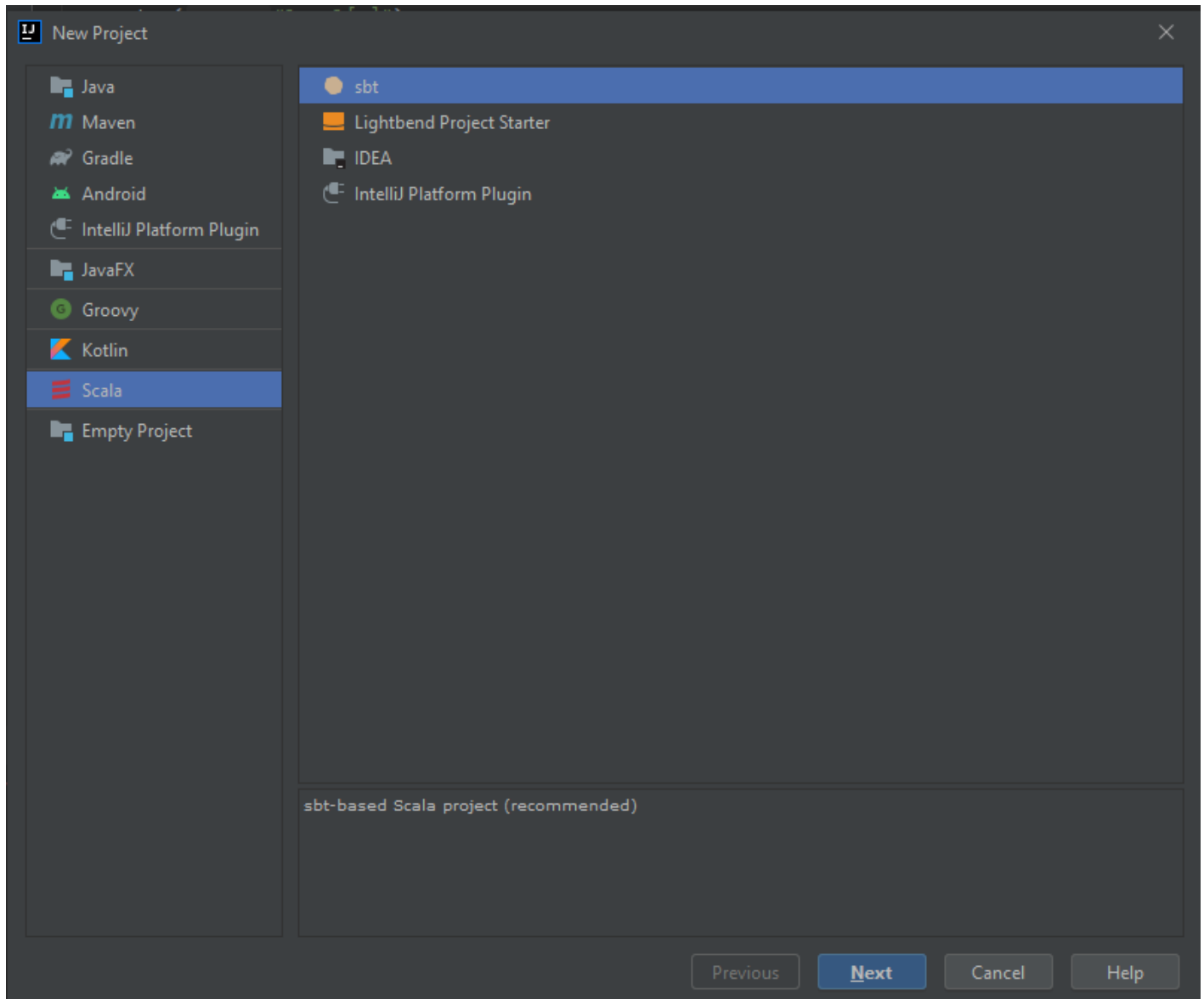
BLOG

KIM JESTEM

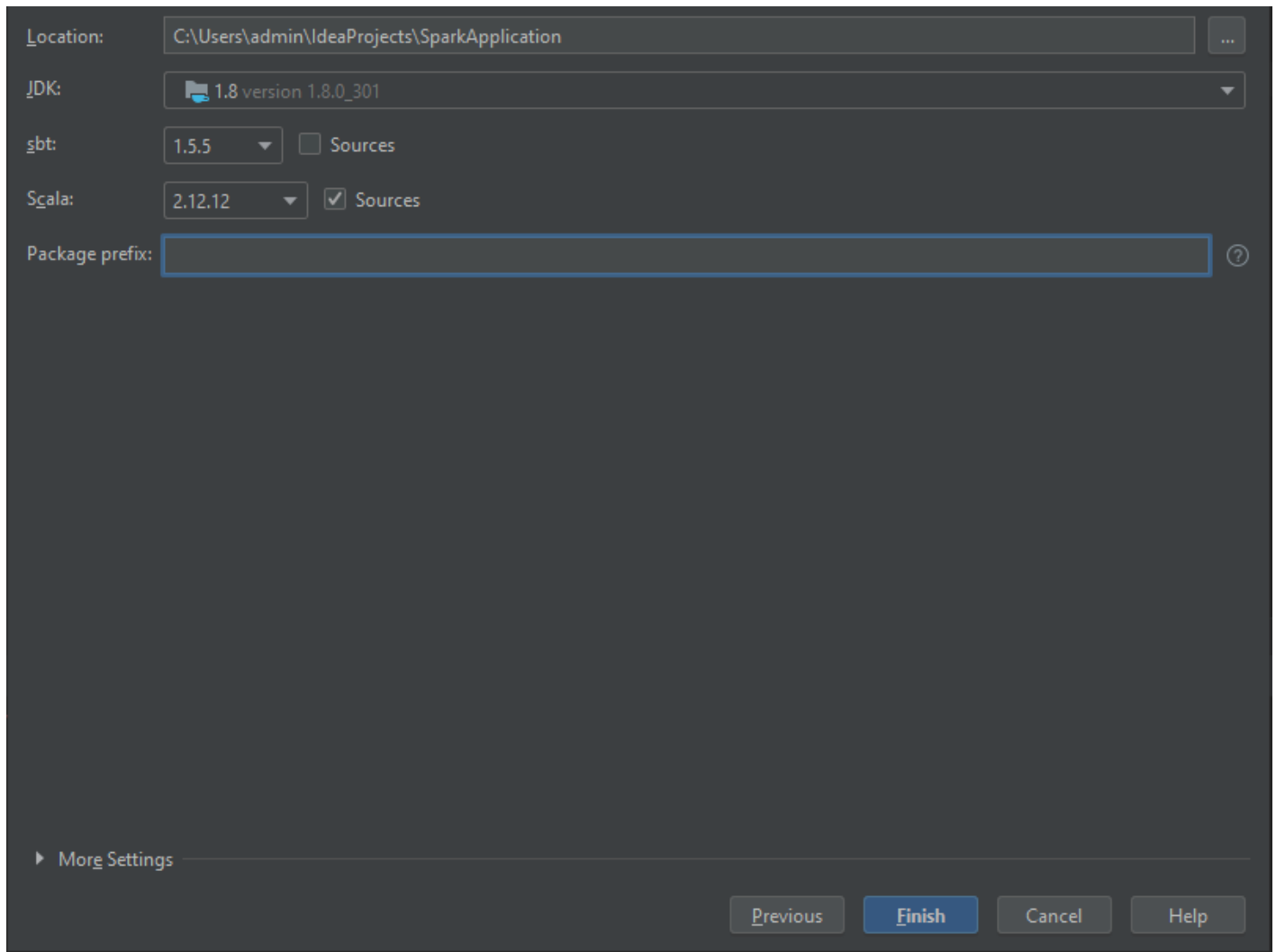
KONTAKT



1. Stwórz nowy projekt. Z okna startowego IntelliJ lub z Menu File > New > Project. Wybierz Scala > sbt



2. Podaj nazwę projektu "<SparkApplication>."

[START](#)[BLOG](#)[KIM JESTEM](#)[KONTAKT](#)

Location: C:\Users\admin\IdeaProjects\SparkApplication

JDK: 1.8 version 1.8.0\_301

sbt: 1.5.5 ☐ Sources

Scala: 2.12.12 ☒ Sources

Package prefix:

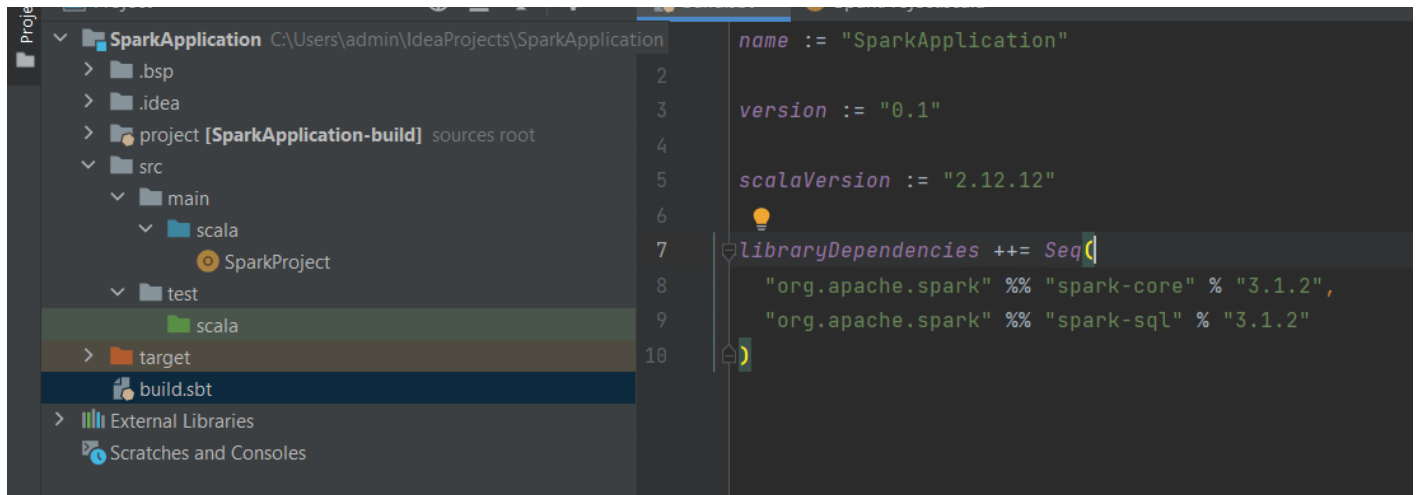
► More Settings

Previous Finish Cancel Help

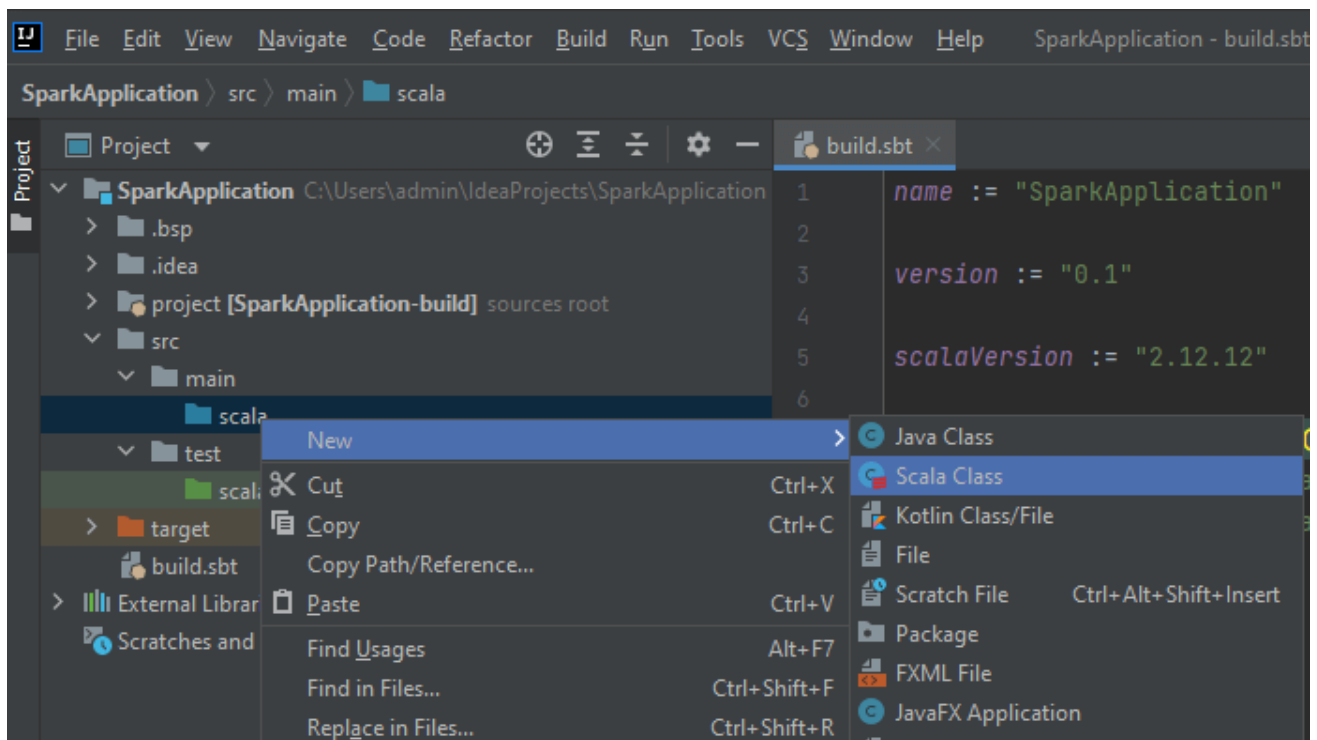
### 3. Kliknij Finish

Otwórz plik **build.sbt** i dodaj biblioteki Sparka. Pamiętaj, że bardzo ważne są wersje wszystkich bibliotek jakie dodasz. Najczęstsze błędy to niekompatybilne wersje języka i/lub wersji bibliotek.

```
1 | libraryDependencies += Seq(  
2 |   "org.apache.spark" %% "spark-core" % "3.1.2",  
3 |   "org.apache.spark" %% "spark-sql" % "3.1.2"  
4 | )
```



#### 4. Dodaj Scala class "SparkProject" jako "Object "



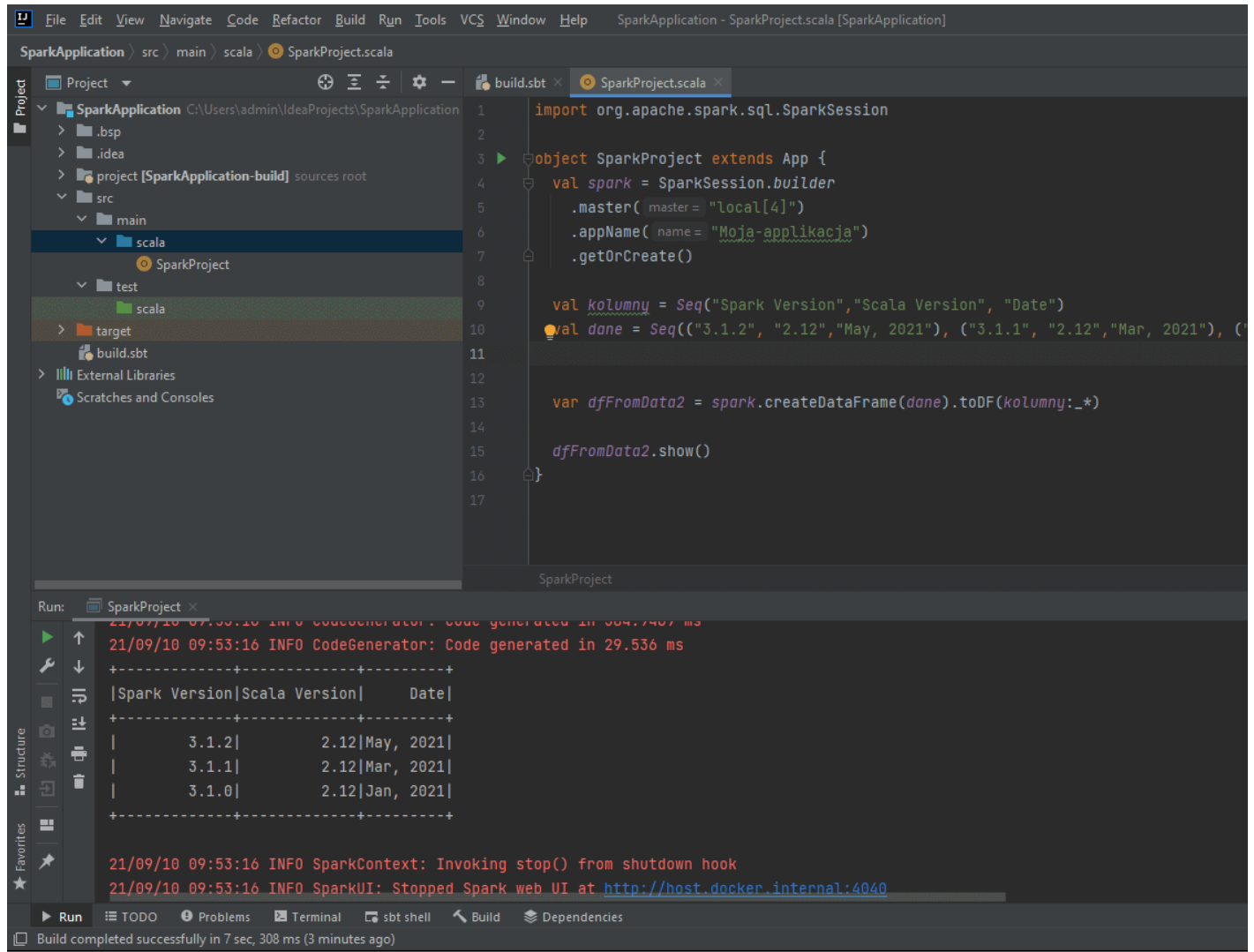
#### 5. Dodaj kod wymagany do uruchomienia Sparka

```

1  import org.apache.spark.sql.SparkSession
2
3  object SparkProject extends App {
4    val spark = SparkSession.builder
5      .master("local[4]")
6      .appName("Moja-aplikacja")
7      .getOrCreate()
8
9    val kolumny = Seq("Spark Version", "Scala Version", "Date")
10   val dane = Seq(("3.1.2", "2.12", "May, 2021"), ("3.1.1", "2.12", "Mar, 2021"),
11
12
13   var dfFromData2 = spark.createDataFrame(dane).toDF(kolumny:_* )

```

6. Klikasz prawym gdziekolwiek w polu tekstowym i wybierasz "Run SparkProject". Jeśli wszystko poszło dobrze to zobaczysz DataFrame. Teraz możesz zacząć eksperymentować ze Sparkiem. Milej zabawy.



The screenshot shows the PyCharm IDE with the SparkProject.scala file open. The code defines a SparkSession, creates a DataFrame from a sequence of data, and displays it. The Run console shows the output of the DataFrame, which is a table with three columns: Spark Version, Scala Version, and Date. The output shows three rows of data.

```
import org.apache.spark.sql.SparkSession

object SparkProject extends App {
  val spark = SparkSession.builder
    .master("local[4]")
    .appName("Moja-aplikacja")
    .getOrCreate()

  val kolumny = Seq("Spark Version", "Scala Version", "Date")
  val dane = Seq(("3.1.2", "2.12", "May, 2021"), ("3.1.1", "2.12", "Mar, 2021"), ("3.1.0", "2.12", "Jan, 2021"))

  var dfFromData2 = spark.createDataFrame(dane).toDF(kolumny: _*)

  dfFromData2.show()
}
```

Run: SparkProject

```
21/09/10 09:53:16 INFO CodeGenerator: Code generated in 384.7407 ms
21/09/10 09:53:16 INFO CodeGenerator: Code generated in 29.536 ms
+-----+-----+-----+
|Spark Version|Scala Version|   Date|
+-----+-----+-----+
|      3.1.2|      2.12|May, 2021|
|      3.1.1|      2.12|Mar, 2021|
|      3.1.0|      2.12|Jan, 2021|
+-----+-----+-----+
21/09/10 09:53:16 INFO SparkContext: Invoking stop() from shutdown hook
21/09/10 09:53:16 INFO SparkUI: Stopped Spark web UI at http://host.docker.internal:4040
```

Build completed successfully in 7 sec, 308 ms (3 minutes ago)

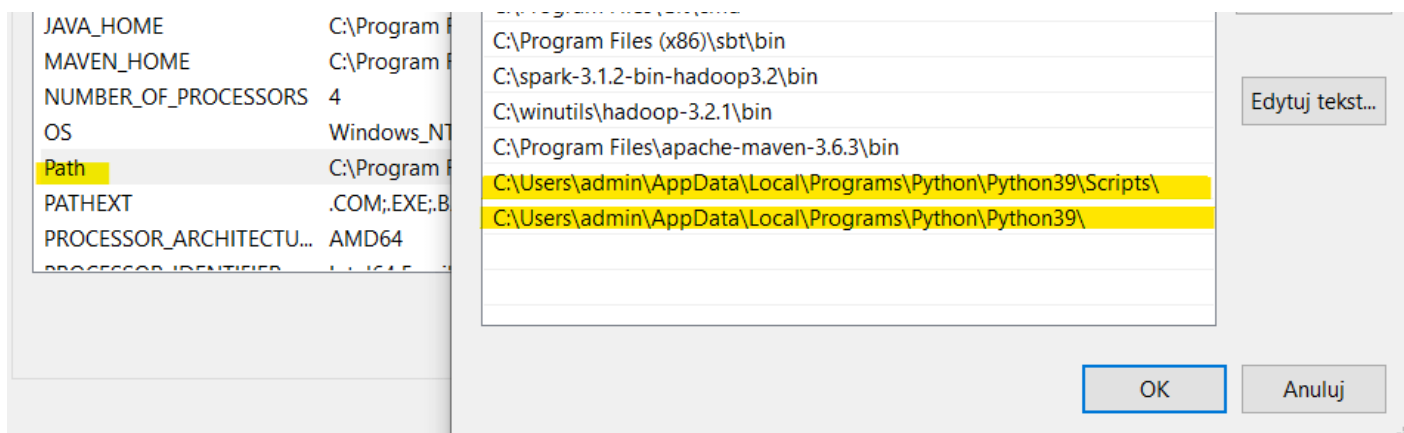
## PyCharm

Jeśli wybrałeś Python to dobrym narzędziem będzie **PyCharm**. Zainstaluj **Community Edition**. Ja zawsze wybieram wszystkie opcje jak coś instaluję żeby się nie ograniczać 😊

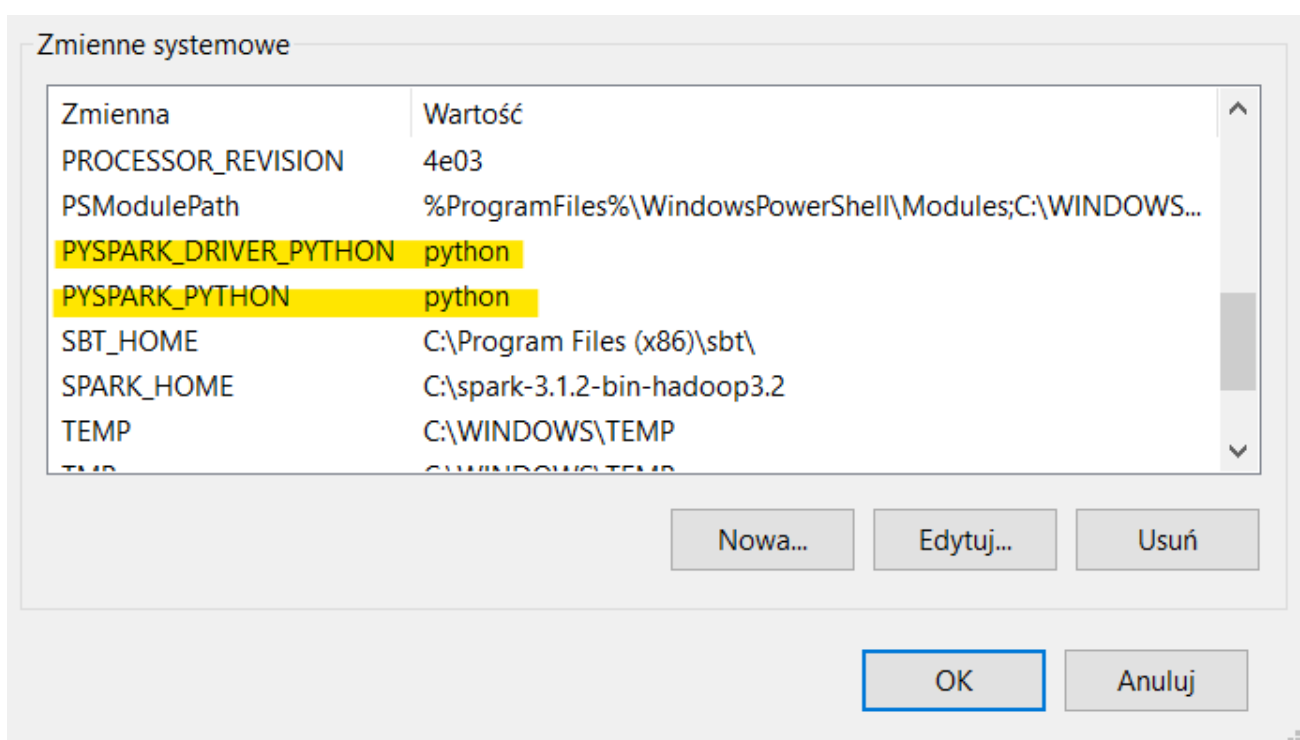
## Konfiguracja dla PyCharm

### Zmienne środowiskowe

1. Upewnij się że ścieżki Python są dodane do systemowych zmiennych środowiskowych








## 2. Dodaj wartości dla zmiennych PYSPARK\_DRIVER\_PYTHON i PYSPARK\_PYTHON



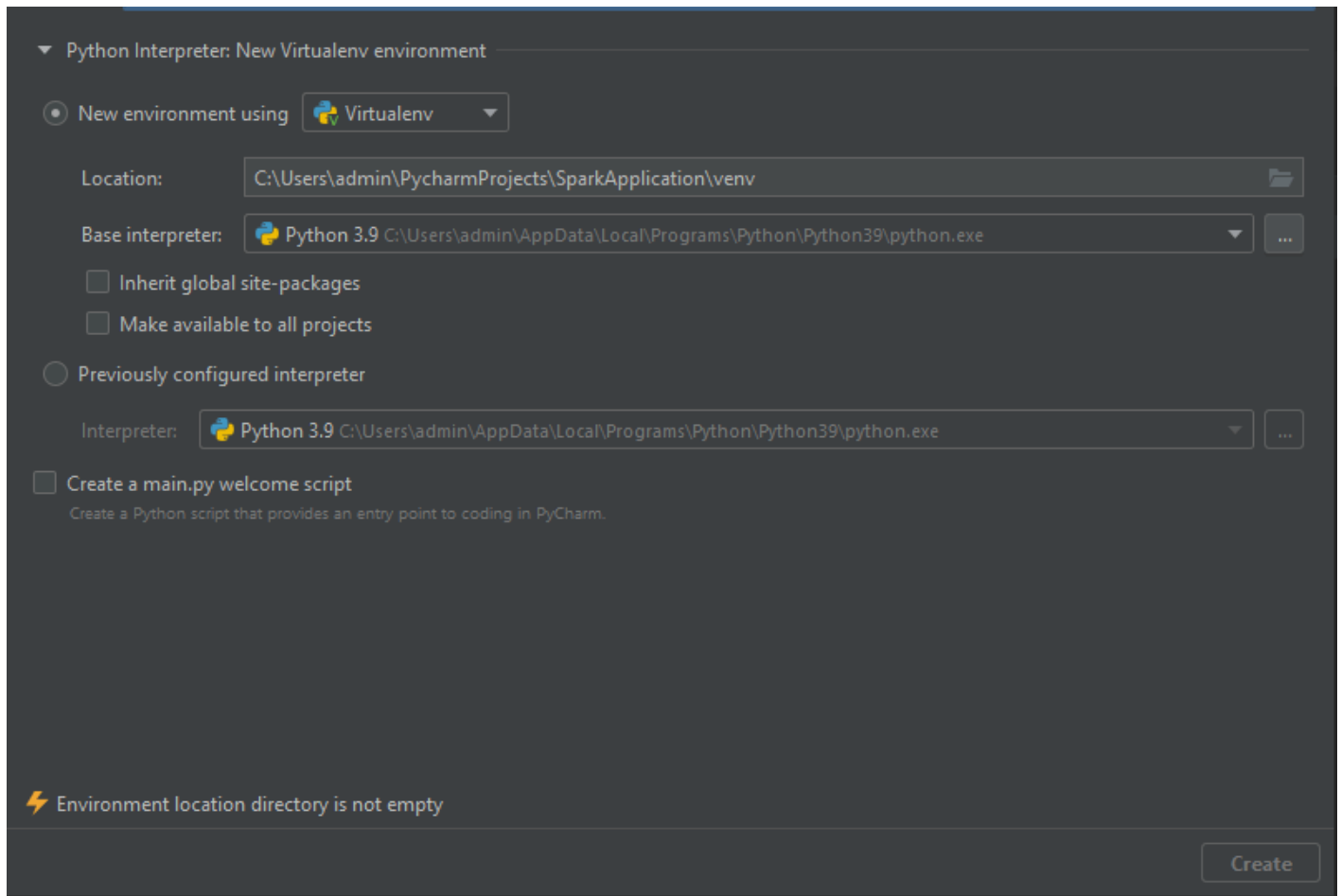
### 3. Wyłącz "Manage App Execution Aliases" (Aliasy wykonania aplikacji) dla Pythona

Aplikacje mogą deklarować nazwę używaną w celu uruchamiania aplikacji z wiersza polecenia. Jeśli wiele aplikacji używa tej samej nazwy, wybierz tę, której chcesz użyć.

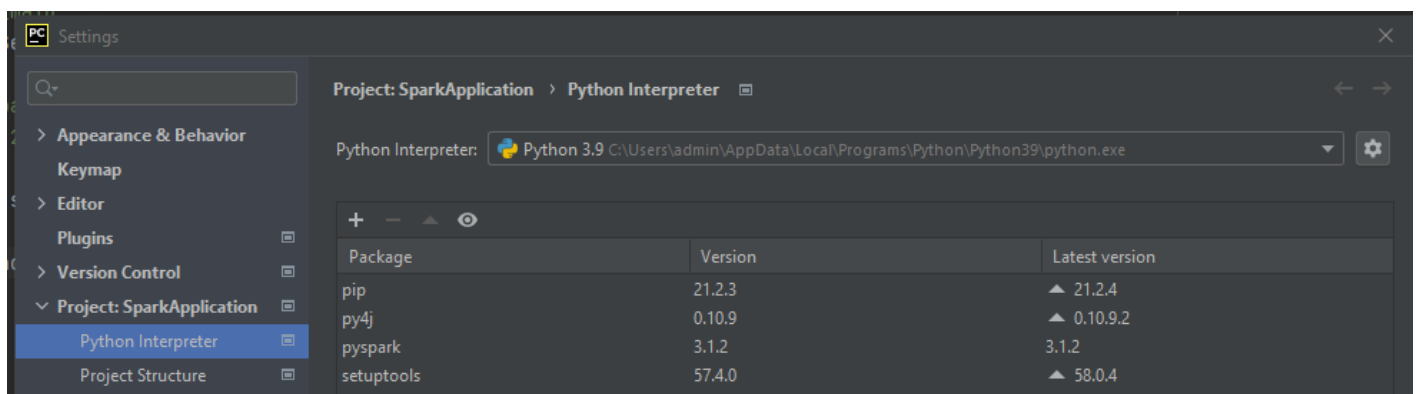
	Skype Skype.exe	<input checked="" type="checkbox"/> Włączone
	Spotify Spotify.exe	<input checked="" type="checkbox"/> Włączone
	Xbox Game Bar GameBarElevatedFT_Alias.exe	<input checked="" type="checkbox"/> Włączone
	Instalator aplikacji python.exe	<input type="checkbox"/> Wyłączone
	Instalator aplikacji python3.exe	<input type="checkbox"/> Wyłączone

1. Stwórz nowy projekt, tutaj powinna wystarczyć tylko nazwa.

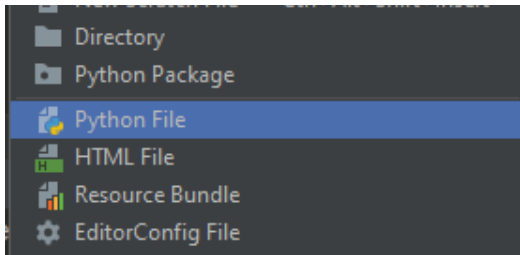


[START](#)[BLOG](#)[KIM JESTEM](#)[KONTAKT](#)

2. Po odpaleniu PyCharm trzeba doinstalować dwie paczki py4j i pyspark. Można to zrobić z File > Settings > Project: <Project Name> – Kliknij znak plus nad tabelką. Wpisujesz py4j a potem pyspark (masz opcję wyboru wersji, upewnij się że pasuje do wersji Spark, którą pobrałeś)



3. Dodaj plik File > New > Python File

[START](#)[BLOG](#)[KIM JESTEM](#)[KONTAKT](#)

#### 4. Dodaj kod

```
1 from pyspark.sql import SparkSession
2
3 if __name__ == '__main__':
4
5     spark = SparkSession.builder.master("local[*]").appName("HelloWorld").getOrCreate()
6
7     kolumny = ["Spark Version", "Scala Version", "Date"]
8
9     dane = [("3.1.2", "2.12", "May, 2021"), ("3.1.1", "2.12", "Mar, 2021"), ("3.1.0", "2.12", "Jan, 2021")]
10
11     dfFromData2 = spark.createDataFrame(dane).toDF(*kolumny)
12
13     dfFromData2.show()
```

Klikasz prawym na Run i jeśli wszystko jest ok to powinieneś widzieć DataFrame.

```
SparkApplication - SparkProject.py
SparkProject.py
1 from pyspark.sql import SparkSession
2
3 if __name__ == '__main__':
4     spark = SparkSession.builder.master("local[*]").appName("HelloWorld").getOrCreate()
5
6     kolumny = ["Spark Version", "Scala Version", "Date"]
7     dane = [("3.1.2", "2.12", "May, 2021"), ("3.1.1", "2.12", "Mar, 2021"), ("3.1.0", "2.12", "Jan, 2021")]
8
9     dfFromData2 = spark.createDataFrame(dane).toDF(*kolumny)
10
11     dfFromData2.show()
```

Run: SparkProject

```
C:\Users\admin\AppData\Local\Programs\Python\Python39\python.exe C:/Users/admin/PycharmProjects/SparkApplication/SparkProject.py
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
+-----+-----+-----+
|Spark Version|Scala Version|    Date|
+-----+-----+-----+
|      3.1.2|      2.12|May, 2021|
|      3.1.1|      2.12|Mar, 2021|
|      3.1.0|      2.12|Jan, 2021|
+-----+-----+-----+

Process finished with exit code 0
```