



University  
of Basel

# Jupyter-Notebooks

Marcel Lüthi, Departement Mathematik und Informatik, Universität Basel

# Jshell: Interaktives Programmieren

```
C:\Users\luetma00>jshell
| Welcome to JShell -- Version 10.0.1
| For an introduction type: /help intro

jshell> System.out.println("hello world");
hello world

jshell>|
```

- Jshell: Interaktives Programmieren in Java
  - Teil von Java seit Java 9
-

# Jshell: Vorteile und Nachteile

## Vorteile

- Kein Schreiben von Klassenrumpf und main Methode nötig
- Kompilieren und Ausführen stark vereinfacht
- Direktes Feedback erlaubt interaktives Arbeiten

*Gut zum experimentieren.*

## Nachteile

- Nur textbasiert
- Editieren von etwas längeren Funktionen ist mühsam
- Keine Historie der Befehle

*Nicht geeignet um Experimente später nachzuvollziehen.*

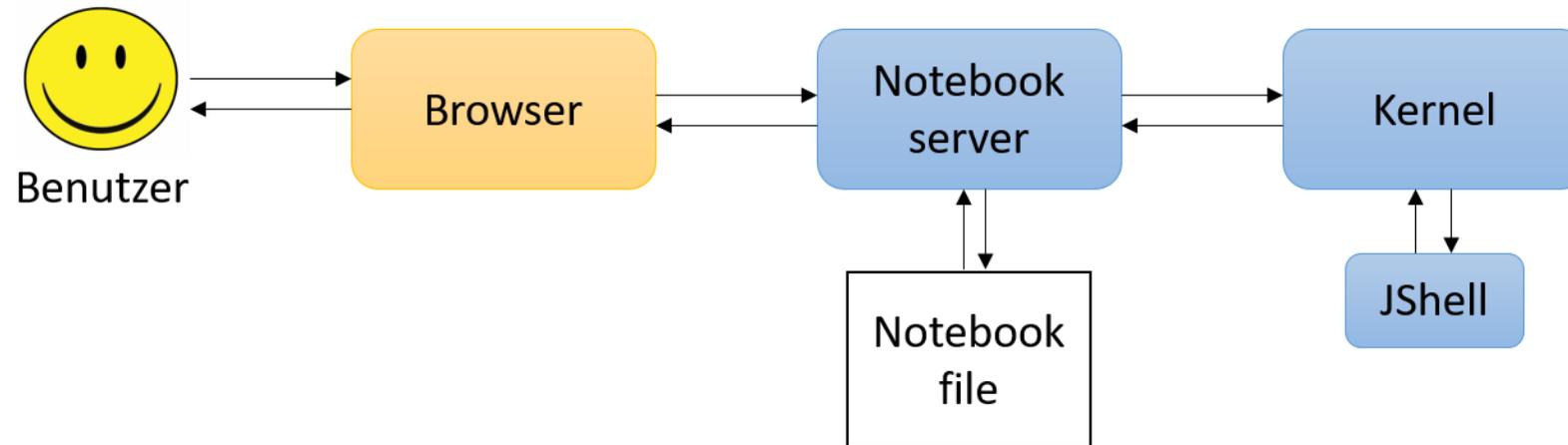
---

# Was sind Jupyter Notebooks

Webbasierte Programmierumgebung

- Idee: Vereinen von Dokumentation und Programme
- Unterstützt viele verschiedene Programmiersprachen.

Java-Kernel: Basierend auf JShell



# Demo

JUPYTER FAQ </> [Menu] [Refresh] [Close] [Download]

gymint-programmieren / notebooks

## Einführung in Jupyter notebooks

In diesem Notebook geben wir eine kleine Einführung in Jupyter Notebooks und zeigen wie man einfache Java Programme schreibt. Ausserdem führen wir Turtlegrafik ein, die wir während dem Kurs immer wieder verwenden werden um Programmierkonzepte zu illustrieren.

### Einfache Java Programme in Jupyter Notebooks

Jupyter Notebooks sind im Dokumente, die neben Text auch Programmcode enthalten aus einer Sequenz von einzelnen Zeilen bestehen. Die aktuelle Zeile erkennt man jeweils daran, dass diese umrahmt ist. Jede Zeile ist entweder eine Textzeile oder eine Code Zeile. Codezeilen erkennt man dadurch, dass diese mit `In [..]` gekennzeichnet sind. Codezeilen enthalten ausführbaren Java Code und können durch drücken von `Shift+Enter` ausgeführt werden. Die folgenden zwei Zeilen sind Codezeilen und können ausgeführt werden.

```
In [8]: 5 + 3;
```

Out[8]: 8

```
In [9]: System.out.println("Willkommen zu Grundlagen der Programmierung")
```

Willkommen zu Grundlagen der Programmierung

**Mini-Übung:**

- Führen Sie die obigen 2 Codezeilen aus.

Wir wir sehen können wir hier im Gegensatz zu richtigen Java Programmen, direkt einzelne Java Ausdrücke eingeben, die dann direkt ausgeführt werden. Wir brauchen kein Klassengerüst und müssen nicht erst den Compiler aufrufen. Jeder Ausdruck wird direkt von Java (genauer dem Programm JShell) interpretiert.

### Turtle grafik

Wir werden in diesem Kurs häufig Turtle Grafik einsetzen um Konzepte zu illustrieren. Man kann sich vorstellen, dass eine Stiftragende Schildkröte sich auf einer Zeichnungsebene bewegt, und dabei eine Zeichnung erstellt. Die Bewegungen der Schildkröte folgen genau unseren Befehlen.

Damit wir Turtle Grafik nutzen können, müssen wir zuerst eine Programmbibliothek in Jupyter laden:

```
In [12]: %mavenRepo bintray https://dl.bintray.com/egp/maven
%maven ch.unibas.informatik:jturtle:0.3
import ch.unibas.informatik.jturtle.Turtle;
```

Danach können wir die Schildkröte mit Java-Befehlen steuern.

```
In [18]: Turtle turtle = new Turtle();
turtle.penDown();
turtle.forward(50);
turtle.turnRight(120);
turtle.forward(50);
turtle.turnRight(120);
turtle.forward(50);
turtle.toImage();
```

Out[18]:

# Jupyter Notebooks: Unser Einsatz

- Interaktive Slides
- Mini-Übungen als interaktives Element
- Dokumentation von Experimenten / Programmen zum individuellen Lernen und zur Prüfungsvorbereitung

## Zuweisungen und arithmetische Funktionen

Durch das Arbeiten mit Variablen können wir Programme allgemeiner schreiben und von konkreten Beispielen abstrahieren. Als Beispiel nehmen wir die Berechnung der Hypotenuse eines rechtwinkligen Dreiecks. Anstatt die Berechnung direkt mit konkreten Werten durchzuführen, können wir Variablen einführen und die Formel dann allgemein hinschreiben.

```
In [15]: double a = 3;
         double b = 4;
```

Nun wissen wir vom Satz von Pythagoras, dass die Länge der Hypotenuse  $c$  mit der Formel  $c^2 = a^2 + b^2$  berechnet werden kann.

```
In [20]: double aSquared = a * a;
         double bSquared = b * b;
         double cSquared = aSquared + bSquared;
```

Um  $c$  zu erhalten müssen wir noch die Wurzel ziehen. Dafür stellt uns Java die Funktion `Math.sqrt` zur Verfügung.

```
In [21]: double c = Math.sqrt(cSquared);
```

Die Werte der Variablen können wir dies mit dem Befehl `System.out.println(variable)` ausgeben lassen.

```
In [22]: System.out.println(c);
         5.0
```

### Mini Übung

- Definieren Sie eine Variable mit dem Namen `z` die nur ganze Zahlen enthalten kann und weisen Sie dieser den Wert 3 zu.
- Was passiert, wenn Sie statt 3 den Wert 3.0 zuordnen?
- Wie müssen Sie den Typ ändern, damit die Zuweisung `z = 3.0` funktioniert?

```
In [24]: // Schreiben Sie Ihre Lösungen hier hin. Neue Zeilen können Sie mit Alt + Enter hinzufügen.
```

## Verzweigung

Im Programmieren muss man oft zwischen verschiedenen Fällen unterscheiden. Dafür können wir die `if else` Anweisung benutzen. Als erstes Beispiel schauen wir uns ein Programm an, das entscheidet ob die Variable `z` einen positiven oder negativen Wert beinhaltet.

```
In [97]: int z = 5;
```

# Jupyter Notebooks: Nachteile

- Entwicklung grösserer Programme nicht möglich
- Keine Unterstützung von professionellen Entwicklungsumgebungen
- Teile von Notebooks können in beliebiger Reihenfolge ausgeführt werden → Verwirrungspotenzial

*Super zum lernen, nicht geeignet zur Entwicklung grosser Programme.*

---

# Installation / Zugriff auf Jupyter-Notebooks



1. Webbasierte Lösung (MyBinder.org)
  - + Keine Installation – Läuft auf allen Geräten
  - + Überall verfügbar
  - + Gratis
  - Notebook muss umständlich lokal gespeichert werden
  - Abhängigkeit von externem Service
2. Lokale Installation
  - + Speicher / Laden einfach
  - + Nutzt lokale Rechenressourcen
  - Installation etwas aufwändig
3. Docker
  - + Speicher / Laden einfach
  - + Nutzt lokale Rechenressourcen
  - + Docker muss installiert sein

- Link zu Installationsanleitung für Docker auf Vorlesungsseite ([bit.ly/gyminf-programmieren](https://bit.ly/gyminf-programmieren))

---

# Aufgabe: Lernen Sie Jupyter Notebooks kennen.

Jupyter nbviewer

JUPYTER FAQ </> [Menu] [Refresh] [Share] [Download]

gyminf-programmieren / notebooks

## Einführung in Jupyter notebooks

In diesem Notebook geben wir eine kleine Einführung in Jupyter Notebooks und zeigen wie man einfache Java Programme schreibt. Ausserdem führen wir Turtlegrafik ein, die wir während dem Kurs immer wieder verwenden werden um Programmierkonzepte zu illustrieren.

### Einfache Java Programme in Jupyter Notebooks

Jupyter Notebooks sind im Grunde Dokumente, die neben Text auch Programmcode enthalten, aus einer Sequenz von einzelnen Zeilen bestehen. Die aktuelle Zeile erkennt man jeweils daran, dass diese umrahmt ist. Jede Zeile ist entweder eine Textzeile oder eine Code Zeile. Codezeilen erkennt man dadurch, dass diese mit 'In [...]' gekennzeichnet sind. Codezeilen enthalten ausführbaren Java Code und können durch drücken von `Shift+Enter` ausgeführt werden. Die folgenden zwei Zeilen sind Codezeilen und können ausgeführt werden.

```
In [ ]: 5 + 3;
```

```
In [ ]: System.out.println("Programmieren macht Spass");
```

**Mini-Übung:**

- Führen Sie die obigen 2 Codezeilen aus.

Wir hier sehen können wir hier im Gegensatz zu richtigen Java Programmen, direkt einzelne Java Ausdrücke eingeben, die dann direkt ausgeführt werden. Wir brauchen kein Klassengertist und müssen nicht erst den Compiler aufrufen. Jeder Ausdruck wird direkt von Java (genauer dem Programm JShell) interpretiert.

### Turtle grafik

Wir werden in diesem Kurs häufig Turtle Grafik einsetzen um Konzepte zu illustrieren. Man kann sich vorstellen, dass eine Stiftragende Schildkröte sich auf einer Zeichnungsebene bewegt, und dabei eine Zeichnung erstellt. Die Bewegungen der Schildkröte folgen genau unseren Befehlen.

Damit wir Turtle Grafik nutzen können, müssen wir zuerst eine Programmbibliothek in Jupyter laden:

```
In [ ]: %mavenRepo bintray https://dl.bintray.com/egg/maven
%maven ch.unibas.informatik:jturtle:0.5
import ch.unibas.informatik.jturtle.Turtle;
```

Danach können wir die Schildkröte mit Java-Befehlen steuern:

```
In [ ]: Turtle turtle = new Turtle();
turtle.penDown();
turtle.forward(50);
turtle.turnRight(120);
turtle.forward(50);
turtle.turnRight(120);
turtle.forward(50);
turtle.toImage();
```

Das System ist sehr einfach zu verstehen. Turtle wurde ursprünglich auch dazu entwickelt, Kindern das Programmieren beizubringen. Mithilfe von Turtle Grafik können jedoch auch sehr komplexe und abstrakte Konzepte graphisch illustriert werden. Das folgende Programm ist ein Beispiel für ein komplexes Programm, welches Sie am Ende dieser Vorlesung ohne Probleme verstehen werden.

Notebook: [bit.ly/gyminf-programmieren](https://bit.ly/gyminf-programmieren) -> Alle Notebooks im Überblick -> [JupyterEinfuehrung.ipynb](#)