

TypeScript als erste Programmiersprache für Medienkonzeption Bachelor (MKB)



STUDIEREN
AUF HÖCHSTEM
NIVEAU

Inhalte entsprechend Modulhandbuch

Programmieren

- Algorithmus
- Informationsrepräsentation
- Datentypen
- Kontrollstrukturen, Methoden
- Objektorientierung
- Vererbung, Polymorphie
- Exceptions, Generics, Enums

GiS

- UCD, Usability, UX
- Responsive-, GUI-, IX-Design
- HTML5, JavaScript
- ISO 9241
- Web-Programmierung
- AJAX
- Datenhaltung / DB-Zugriff

Aktuelle Situation im Grundstudium MBK

Programmieren

Java

mit Processing, dann Eclipse

GIS

JavaScript, PHP

mit Eclipse/Aptana

Warum drei Programmiersprachen und zwei Editoren im Grundstudium MKB ?

Vorteile von Java und Javascript

Java

- Geräte mit JVM/Plugin
- „klassisch“ objektorientiert
- Android nativ
- explizit und statisch typisiert
- Große Softwareprojekte

JavaScript

- Browser nativ
- „Sprache des Web“
- Apps z.B. mit PhoneGap
- Verbund mit HTML(5) & CSS
- Breites Anwendungsfeld

Vorteile Processing und Eclipse/Aptana

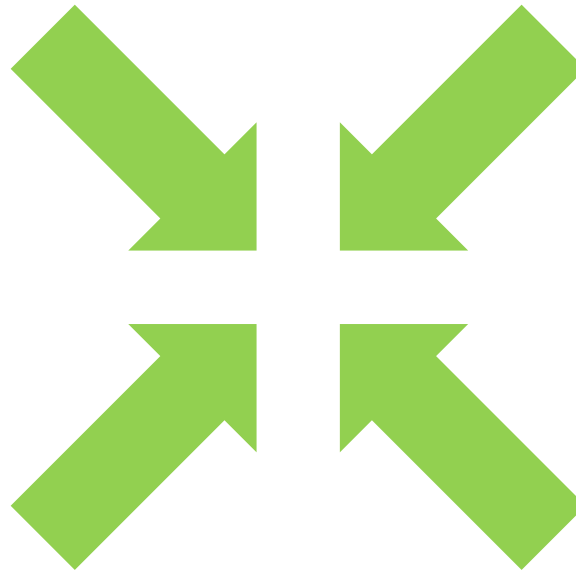
Processing

- intuitiv und einfach
- Erfolge bereits mit wenig Code
- Arbeit mit Grafik von Anfang an
- Export mit Quellcode in HTML

Eclipse/Aptana

- bekannte Standardumgebung
- sehr gute Arbeitsunterstützung
- direkte Webspaceanbindung
- nutzbar für viele Sprachen

Reduktion bei Erhalt der Vorteile möglich?



- Superset von JavaScript
- explizit und statisch typisiert (+ ein dynamischer Typ)
- Klassen, Interfaces, Vererbung, Module, Enums, Generics, etc.
- Zugriffsmodifikation, statische Klassen und Methoden
- greift ECMA6-Standard vor und geht darüber hinaus
- erzeugt JavaScript-Code und ist zu diesem kompatibel
- entwickelt für große, komplexe und skalierbare Web-Applikationen
- Unterstützung in Eclipse/Aptana
- Microsoft, Anders Hejlsberg (Turbo Pascal, Delphi, J++, C#, .NET)

Zukünftige Situation im Grundstudium MBK

Programmieren

TypeScript

mit Eclipse/Aptana
im TypeScript-View
und mit einem Setup-Script

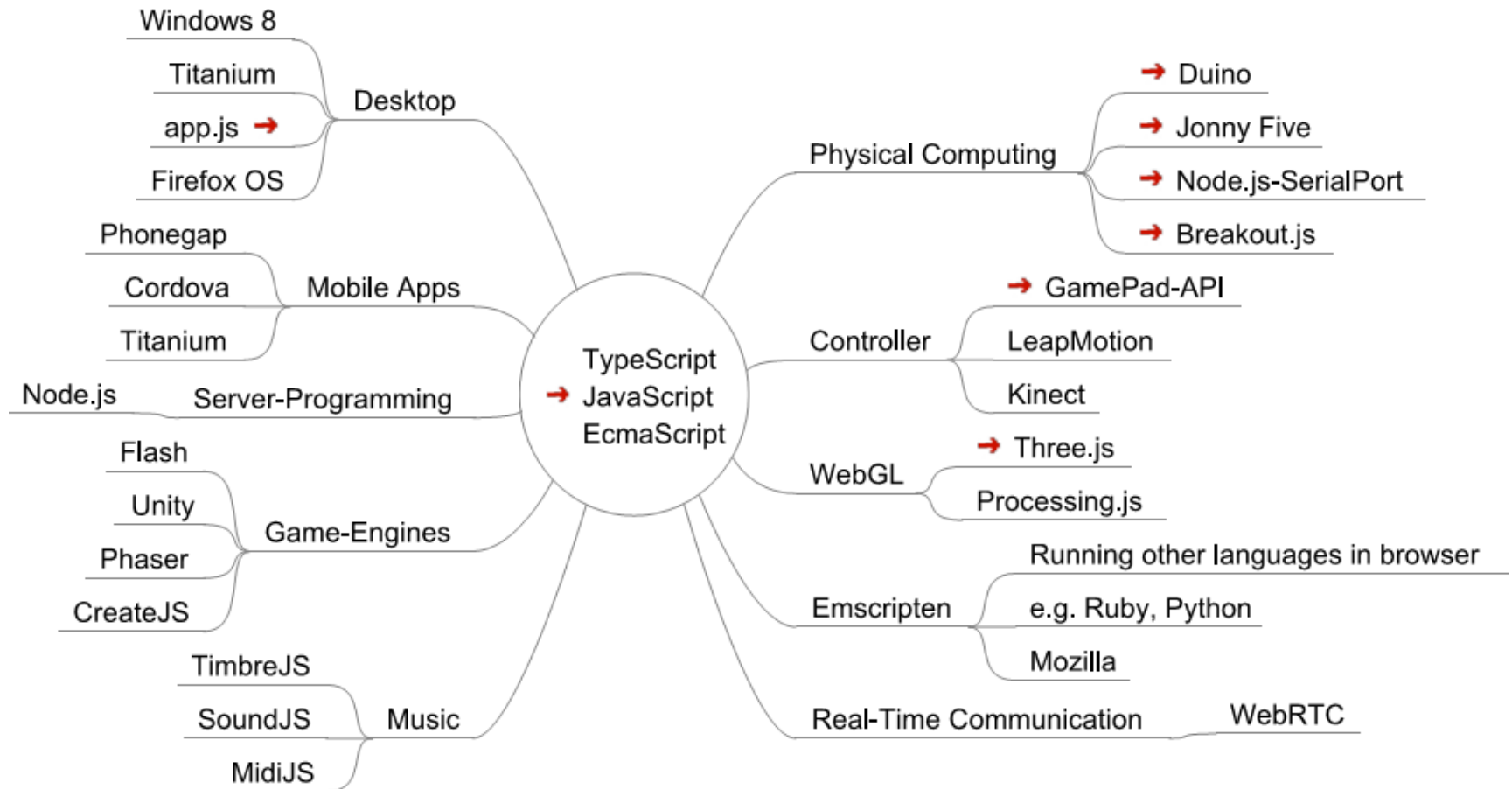
GIS

TypeScript, PHP

mit Eclipse/Aptana
„downgrade“ zu JS möglich
(aber nicht empfohlen...)

- Kombination vieler Vorzüge für Lehre und Praxis
- Zwei Semester mit einer strukturierten, objektorientierten Sprache
- Weitere im Hauptstudium auf dieser Basis bei Bedarf leichter erlernbar
- TypeScript für Praxis in MKB mehr als ausreichend
- Einbindung von Bibliotheken (z.B. JQuery) inklusive Typechecking
- Weite Betätigungsfelder (da JavaScript)

Betätigungsfelder



Praxisbeispiel



Anhang: Codebeispiele

STUDIERN
AUF HÖCHSTEM
NIVEAU

Beispielvergleich „Hello World“

Java


```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello, World");  
    }  
}
```

JavaScript

```
console.log ("Hello World");
```

Beispielvergleich „Fehlerhafter Code“

Java



```
//Fehlerhafter Code
String t = "test";
t.prop = 100;
System.out.print(2 * t);
```

prop cannot be resolved or is not a field

The operator * is undefined for the argument type(s) int, String

JavaScript

```
1 //Fehlerhafter Code
2 var t = "test";
3 t.prop = 100;
4 console.log(2 * t);
```

Keine Fehleranzeige, weder zur Schreib- noch zur Laufzeit

Die Beispiele in TypeScript

```
1 console.log("Hello World");  
2  
3 //Fehlerhafter Code  
4 var t = "test";  
5 t.prop = 100;  
6 console.log(2 * t);
```

Property 'prop' does not exist on type 'string'.

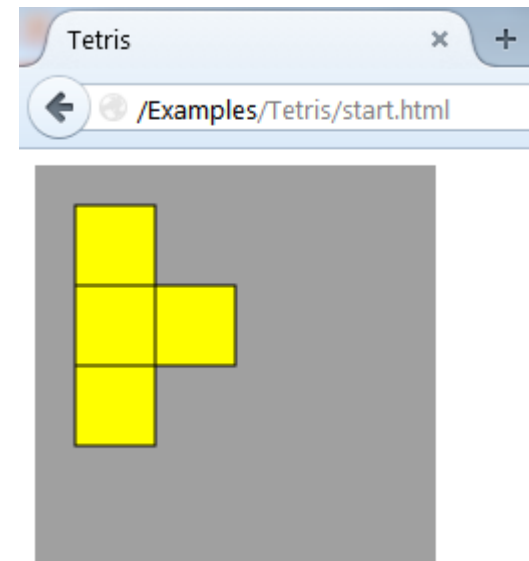
The right-hand side of an arithmetic operation must be of type 'any', 'number' or an enum type.

Einstieg in TypeScript

Code

```
Setup.title("Tetris");  
Setup.size(200, 200);  
painter.fillStyle = "#FFFF00";  
painter.rect(20,20,40,40);  
painter.rect(20,60,40,40);  
painter.rect(60,60,40,40);  
painter.rect(20,100,40,40);  
painter.fill();  
painter.stroke();
```

Ergebnis




```
1 module LanchestersLaw {
2   enum PARTY { NEUTRAL, RED, BLUE };
3   import V = Vector2D;
4
5   export class Unit extends Moveable implements Target {
6     private position: V.Vector2D = new V.Vector2D(0, 0);
7     private party: PARTY = PARTY.NEUTRAL;
8     private health: number = 1.0;
9     private color: string = "#000000";
10    private posTargetX: number;
11    private posTargetY: number;
12
13    constructor(_party: PARTY, _position: V.Vector2D) {
14      super(_position);
15      this.setParty(_party);
16    }
17
18    public findNearestUnit(_aUnits: Unit[], _party: PARTY): Unit {
19      var maxDistance = Infinity;
20      var closest: Unit = null;
21      for (var i: number = 0; i < _aUnits.length; i++) {
22        var test: Unit = _aUnits[i];
23        if (test.health == 0 || test.party != _party || this == test)
24          continue;
25        var d: number = this.position.getDistanceTo(test.position);
26        if (d < maxDistance) {
27          closest = test;
28          maxDistance = d;
29        }
30      }
31      return closest;
32    }
33  }
```

Fenton, Steve: „Pro TypeScript“. Apress, 2014

„TypeScript“, <http://www.typescriptlang.org/>. Microsoft, 12/2014

Anger, Karjoth, Haas, Berger, Rebmann: „Garage Tales“. HFU 2014

ENDE

STUDIEREN
AUF HÖCHSTEM
NIVEAU