Dart

David Stutz

infuhrung

Motivation

Werkzei

Demonstration

Dart

Klassen Optionale Typer DOM Modularität

Evaluation

Quellen

usatz

Typ-Überprüfung Optionale Paramete Klassen

Dart

Eine kurze Einführung

David Stutz

Lehrstuhl für Datenmanagement und -exploration RWTH Aachen

Proseminar SS 2012

Uberblick

- Motivation
- 3 Werkzeuge
- Demonstration
- Dart
 - Klassen
 - Optionale Typen
 - DOM
 - Modularität
- **Evaluation**
- Quellen
- 8 Zusatz
 - Typ-Überprüfung
 - Optionale Parameter
 - Klassen

1

Typ-Überprüfung

DART

- Klassen-basierte Programmiersprache für Webanwendungen
- Entwickelt von Google

¹Bild: http://www.dartlang.org/

Einführur

Motivation

vverkzeuge

Dart

Optionale Typen

Evaluatio

Quellen

usatz

Typ-Überprüfung
Optionale Parameter

DART

1

- Klassen-basierte Programmiersprache für Webanwendungen
- ► Entwickelt von Google
- Vorgestellt auf der GOTO Konferenz in Aarhus am 12.
 Oktober 2011

¹Bild: http://www.dartlang.org/



Motivation

VVCINZCUGC

Dart

Klassen Optionale Typen DOM

Lvardati

Quellen

usatz

Typ-Überprüfung Optionale Parameter Klassen

DART

1

- Klassen-basierte Programmiersprache für Webanwendungen
- ► Entwickelt von Google
- Vorgestellt auf der GOTO Konferenz in Aarhus am 12.
 Oktober 2011
- Zurzeit: "technical preview"

¹Bild: http://www.dartlang.org/ ←□ト←②ト←②ト←②ト → ③ → ○ ○ ○



1

- Klassen-basierte Programmiersprache für Webanwendungen
- Entwickelt von Google
- Vorgestellt auf der GOTO Konferenz in Aarhus am 12.
 Oktober 2011
- Zurzeit: "technical preview"
- Diese Präsentation: Dart für clientseitige Programmierung

_iiii uiii uiig

Motivation

Werkzeuge

Demonstration

Dart

Optionale Typen
DOM

_varaacic

⊋uellen

ısatz

¹Bild: http://www.dartlang.org/ ←□ → ←② → ←② → ←② → → ② → ○ ○

Motivation

Demonstration

Dart

Klassen
Optionale Typen
DOM
Modularität

.vaiuatio

uellen

ısatz

Typ-Überprüfung Optionale Parameter Klassen

"Building delightful applications on the web today is far

too difficult." 2 - Mark S. Miller

²https://gist.github.com/1208618 □ → ← 🗗 → ← 🖫 → 👢 → ९९ 🦠

Motivation

Werkzeuge

Demonstration

Dart

Klassen Optionale Typen DOM

- varuatio

Quellen |

satz

Typ-Überprüfung Optionale Parameter Klassen

► Komplexe Anwendungen benötigen modularen und strukturierten Code

^{► &}quot;Building delightful applications on the web today is far too difficult." ² – Mark S. Miller

²https://gist.github.com/1208618 □ → ⟨♂ → ⟨ ≧ → ⟨ ≧ → ⟨ ≧ → ⟨ ≥ → ⟨ ⊘ へ ⊘

Motivation

rrentzeage

Demonstration

Dart

Klassen Optionale Typen DOM Modularität

_vaiuatioi

uellen

usatz

Typ-Überprüfung Optionale Paramete Klassen

► "Building delightful applications on the web today is far too difficult." ² – Mark S. Miller

- Komplexe Anwendungen benötigen modularen und strukturierten Code
- ▶ Kaum Möglichkeiten homogene Systeme zu entwickeln

²https://gist.github.com/1208618 □ > ← 🗗 > ← 🛢 > → 🛢 → 🔾 ↔ 🤇

Motivation

rrentzeage

Demonstration

Dart

Klassen Optionale Typen DOM

Lvaiuatio

uellen

usatz ..

Typ-Überprüfung Optionale Paramete Klassen

 "Building delightful applications on the web today is far too difficult." ² – Mark S. Miller

- Komplexe Anwendungen benötigen modularen und strukturierten Code
- ▶ Kaum Möglichkeiten homogene Systeme zu entwickeln
 - Google Web Toolkit https://developers.google.com/web-toolkit/
 - Node.js http://nodejs.org/

²https://gist.github.com/1208618 □ > ← 🗗 > ← 🛢 > → 🛢 → 🔾 ↔ 🤇

Motivation

vverkzeuge

Demonstration

Dart

Klassen Optionale Typen DOM

valuatioi

uellen

usatz

- "Building delightful applications on the web today is far too difficult." ² – Mark S. Miller
- ► Komplexe Anwendungen benötigen modularen und strukturierten Code
- ▶ Kaum Möglichkeiten homogene Systeme zu entwickeln
 - Google Web Toolkit https://developers.google.com/web-toolkit/
 - Node.js http://nodejs.org/
- Clientseitig vorallem Schwierigkeiten mit Javascript

²https://gist.github.com/1208618 □ → ⟨♂ → ⟨ ≧ → ⟨ ≧ → ⟨ ≧ → ⟨ ≥ → ⟨ ⊘ へ ⊘

intuhrung

Motivation

Demonstration

Dart

Optionale Typen
DOM

.vaiuatio

uellen

satz

Typ-Überprüfung Optionale Parameter Klassen

Google stellt hilfreiche Werkzeuge zur Entwicklung bereit:

Dart–zu–Javascript Übersetzer

David Stutz

inführung

Motivation

Verkzeug

Demonstration

5

Klassen

Optionale Typen DOM

odularität

Quellen

usatz

Typ-Überprüfung
Optionale Paramete

Google stellt hilfreiche Werkzeuge zur Entwicklung bereit:

- ► Dart-zu-Javascript Übersetzer
- ▶ Dart Editor auf Eclipse basierter Editor

Motivation

Verkzeu

Demonstration

Google stellt hilfreiche Werkzeuge zur Entwicklung bereit:

- ► Dart-zu-Javascript Übersetzer
- Dart Editor auf Eclipse basierter Editor
- "Dartium" auf Chromium basierter Browser mit integrierter Dart VM

Dart

Klassen
Optionale Typen
DOM

lodularität

Lvaidati

Quellen

satz

David Stutz

inführung

Motivation

Verkzeug

Demonstration

Google stellt hilfreiche Werkzeuge zur Entwicklung bereit:

- ► Dart-zu-Javascript Übersetzer
- Dart Editor auf Eclipse basierter Editor
- "Dartium" auf Chromium basierter Browser mit integrierter Dart VM

Downloads unter http://dartlang.org.

Dart

Klassen

optionale Typ OM

lodularität

Quellen

satz

Motivation

Demonstration

Dart

Klassen Optionale Typen DOM

Evaluati

Quellen

usatz

Typ-Überprüfung Optionale Parameter Klassen

Demonstration

► Klassen-basierte Objektorientierung

-iiii uiii uiig

Motivation

Werkzeuge

Demonstratio

Dart

Optionale Typen OM Iodularität

valuatio

Quellen

satz

- ► Klassen-basierte Objektorientierung
- ► Einfache Vererbung

Einführung

Motivation

Werkzeuge

Demonstratio

Dart

Optionale Typen DOM

valuatio

uellen

usatz

- Klassen-basierte Objektorientierung
- ► Einfache Vererbung
- Abstrakte Klassen

EINTUNTUNG

Motivation

Dart

Optionale Typen DOM

valuation

uellen

satz

- Klassen-basierte Objektorientierung
- ► Einfache Vererbung
- Abstrakte Klassen
- Interfaces

Eintuhrung

Motivation

Demonstratio

Dart

Optionale Typen DOM

valuation

Quallan

uellen

ısatz

- Klassen-basierte Objektorientierung
- ► Einfache Vererbung
- Abstrakte Klassen
- Interfaces

Außerdem:

- Verzicht auf public / private
- ► Getter/Setter mittels get/set deklarieren
- Benannte Konstruktoren

Limumung

Motivation

Verkzeuc

Demonstratio

Dart

Optionale Typen

_varaacic

Quellen

usatz

Optionale Typen

Dart

David Stutz

Eliniunrung

Motivation

Verkzeuge

Demonstration

Dart

Klassen

DOM

valuation

Quellen

Zusatz

Typ-Überprüfung Optionale Paramete Klassen

Eingebaute Typen:

- ► Zahlen num
- ► Zeichenketten String
- ▶ "Collections" Listen, Assoziative Felder...

Lillulliulig

Motivation

Werkzeuge

Demonstratio

Dart

Klassen

ром

Modularität

Quellen

Zusatz

Typ-Überprüfung Optionale Paramete

Eingebaute Typen:

- ► Zahlen num
- ► Zeichenketten String
- "Collections" Listen, Assoziative Felder...

Aber:

- ► Typen nicht verpflichtend
- ► Variablen alternativ mit var deklarierbar

Motivation

Werkzeuge

Demonstration

Dart

Klassen

/lodularität

_varua cro

uellen

satz

Typ-Überprüfung Optionale Parameter Klassen

DOM Manipulation - Zugriff auf HTML:

- ▶ Neue DOM Elemente erstellen:
 - Element.tag('tag')
 - Element.html('<html>...</html>')

intuhrung

Motivation

vverkzeuge

Demonstration

Dart

Klassen

ladularität

Evaluatio

Quallan

satz

Typ-Überprüfung Optionale Parameter Klassen

DOM Manipulation - Zugriff auf HTML:

- ▶ Neue DOM Elemente erstellen:
 - Element.tag('tag')
 - ► Element.html('<html>...</html>')
- ▶ Auf Attribute direkt mittels element. attributes zugreifen

Motivation

vverkzeuge

Demonstration

Dart

Klassen

lodularität

Lvardatic

Quellen

ısatz

Typ-Überprüfung
Optionale Parameter

DOM Manipulation - Zugriff auf HTML:

- ▶ Neue DOM Elemente erstellen:
 - Element.tag('tag')
 - ► Element.html('<html>...</html>')
- ▶ Auf Attribute direkt mittels element. attributes zugreifen
- ► Knoten von DOM Elementen mittels element.nodes ansprechen

WIOLIVALIOII

Demonstratio

.

Klassen Optionale Typen

valuation

uellen

usatz

Typ-Überprüfung Optionale Parameter Klassen

Anwendungen lassen sich modular aufbauen:

► Eigene Bibliotheken erstellen:

```
1 // In View.dart eine Bibliothek definieren :
2 #library('view');
```

Motivation

vverkzeuge

Demonstration

Dart

Klassen
Optionale Typen
DOM

valuation

Quellen

Zusatz

Typ-Überprüfung Optionale Parameter Klassen

```
Anwendungen lassen sich modular aufbauen:
```

► Eigene Bibliotheken erstellen:

```
1 // In View.dart eine Bibliothek definieren :
2 #library('view');
```

Bibliotheken einbinden:

```
1 // Dart's HTML Bibliothek einbinden:
2 #import('dart:html');
3 // Eigene Bibliothek einbinden:
4 #import('View.dart');
```

Vererbung

Bibliotheken

CSS Selektoren

Optionale Typen

Dart

Typ-Überprüfung

Javascript

Klassen-basiert, Interfaces, Prototypen-basiert, komplizierte Vererbung erschwerte Typüberprüfung

- "undefined"
- "falsify"

Keine Bibliotheken

- Globaler Namespace
- externe Bibliotheken (z.B. JQuery³)

DOM Manipulation mittels

³http://jquery.com/

, i

)emonstratio

Dart

Klassen Optionale Ty DOM

odularität

valuation

ellen

ısatz

Typ-Überprüfung Optionale Paramete Klassen

Probleme für den clientseitigen Einsatz:

Übersetzung zu Javascript vor Ausführung

⁴http:

 $/\!/ en.wikipedia.org/wiki/Dart_(programming_language) \qquad {\tt ?} \quad {\it ?} \bigcirc {\it ?} \\$

Motivation

vverkzeuge

Demonstration

Dart

Klassen Optionale Typen DOM

uellen

usatz ..

Typ-Überprüfung Optionale Parameter Klassen

Probleme für den clientseitigen Einsatz:

- ▶ Übersetzung zu Javascript vor Ausführung
- Nativer Browser-Support − "as of March 2012, Microsoft Internet Explorer, Mozilla Firefox, Opera Software's Opera browser, and Apple Safari do not have plans to implement support for Dart" ⁴

//en.wikipedia.org/wiki/Dart_(programming_language) > 💈 🔊 🤉 🗟

⁴http:

David Stutz

Einführung

Motivation

vverkzeuge

Demonstration

Dart

Klassen Optionale

> OOM Modularität

Quellen

ısatz

Typ-Überprüfung
Optionale Parameter

Dartlang.org
http://dartlang.org

Dart (programming language)
http://en.wikipedia.org/wiki/Dart_
(programming_language)

Dart: 10 Punkte, in denen es JavaScript übertrifft http://t3n.de/news/ dart-10-punkte-denen-javascript-358345/

Notivation

verkzeuge

)emonstratio

Dart

Klassen
Optionale Typen
DOM

Evaluat

Quellen

usatz

Typ-Überprüfung
Optionale Parameter
Klassen

Vielen Dank für die Aufmerksamkeit!

Fragen?

Typ-Überprüfung

Dart

David Stutz

```
var nullVal = null;
if (! nullVal) {
    // 'null' wird als 'false'
    behandelt ...
4 }
```

intuhrung

Motivation

Wernzeage

o cinonistratio

Klassen
Optionale Typer

valuation

uellen

usatz

Optionale Parameter Klassen

Motivation

Werkzeuge

Demonstration

Dart

lassen Optionale Typen OOM Iodularität

Evaluation

uellen

Zusatz

Optionale Parameter Klassen

```
1 var nullVal = null:
                                    1 var nullVal = null;
2 if (! nullVal ) {
                                    _2 if (nullVal == null) {
3 // 'null' wird als 'false'
                                    3 // Dart kennt lediglich das
         behandelt ...
                                            'wahre' false!
                                    4 }
var emptyString = '';
                                    var emptyString = '';
2 if (!emptyString) {
                                    2 if (emptyString.isEmpty()) {
3 // leere Zeichenketten werden
                                    3 // ...
         als 'false' behandelt ...
4 }
1 var undefinedVal:
2 if (!undefinedVal) {
3 // 'undefined' wird auch als 1 // Dart kennt kein 'undefined'!
         ' false ' behandelt
```

Motivation

Demonstration ()

Dart

(lassen Optionale Type OOM Modularität

.vaiuatio

icotz

Optionale Parameter Klassen

```
Einführung
```

Motivation

vverkzeuge

Demonstratio

```
art
Classen
Optionale Typen
OOM
```

Lvaluatio

Quellen

Zusatz Typ-Überprüfung

lassen

```
1 // Rueckgabewert in der
        Deklaration anzugeben ist
        optional:
 2 foo(x, y, z) {
     return z:
 4 }
 6 foo(1); // NoSuchMethodException!
 8 // y und z sind optionale
        Parameter:
  bar(x, [y, z]) {
     return z:
11
12
13 bar(1); // Wird 'null'
        zurueckgeben!
```

```
1 // Ein Objekt erstellen :
  function Car(brand) {
    this. brand = brand:
                                      5
     Dem Prototypen eine Methode
       hinzufuegen:
  Car.prototype.alertBrand =
       function() {
    alert (this.brand);
```

```
class Car {
     // Durch '_'-Praefix automatisch
           privat:
     var _brand:
     // Konsturktor wird _brand
          automatisch zuweisen:
     Car(this._brand);
     alertBrand() {
       window.alert(this._brand);
12 }
```

Typ-Überprüfung