# Dart

Die revolutionäre Alternative zu JavaScript

# Warum Dart ?

# Warum Dart ?

JavaScript sucks...

- keine (echten) Module
- ===
- keine (echten) Klassen
- VirtualPet.prototype.setCalories = function (newCalories) {...}
- function level scoping
- (function(_obj{useItLater(_obj)})(obj);
- Keine Typ Sicherheit ( static type checking)
- Schwer zu optimieren

# JavaScript sucks...

- Vererbung

```
Base.call(this);
Derived.prototype = Object.create(Base.prototype);
```

WTF...

- Dokumentation generieren ist umständlich ( jsdoc )
- Wenig intelligente Tools ( IDE, refactoring )
- Keine simplen for-each loops für arrays oder dictionaries

# and more...

"Javascript has fundamental flaws that cannot be fixed merely by evolving the language"

http://www.moock.org/lectures/troublewithjs

für weitere Beispiele...

# Dart - a short bio

- Präsentiert bei der GOTO conf in Aarhus, Oct. 2011
- Entwickelt von Lars Bak (und Kaspar Lund)
  - HotSpot Java VM
  - V8 JavaScript VM ( Chrome )
  - 18 Software Patents
  - original Name war mal Spot
- Gilad Bracha (Newspeak)
  - Java Lang und VM Spec designer
  - Beide arbeiteten an Strongtalk
- Kasper Lund
  - Crankshaft introduced adaptive optimizations driven by type-feedback to V8

# Let's look at beautiful Dart

```
abstract class Shape {
  num perimeter();
}

class Rectangle implements Shape {
  final num height, width;
  Rectangle(this.height, this.width);    // Compact constructor syntax.
  num perimeter() => 2*height + 2*width; // Function shorthand syntax.
}

class Square extends Rectangle {
  Square(num size) : super(size, size);
}
```

# Let's look at Dart and HTML

```
import 'dart:html';

main() {
  query('#button-text')
    ..text = "Click me!"
    ..onClick.listen(respond);
}

respond(event) {
  query("#msg").text = "You clicked the button!";
}
```

# Let's talk about Dart

- Optional Types
  - auto completion
  - refactoring
  - compile time error checking
  - documentation
  - and yeah, it's optional :)
  - easy transition from prototype to app
- Classes
  - implicit interfaces
  - class Node extends Spatial
- Lexical scoping
  - Even in a closure within a loop

# Let's talk about Dart libraries

- import "mylibrary.dart"

-> everything there is available, not just one object like with require.js

- part "mylittlepart.dart"

-> externalize small code bits, full visibility

- _thisIsPrivate ( var or method )
- dart2js uses Tree shaking

# Little things

- top-level functions
- [optional=1] and {named:2} parameters
- "string interpolation: $myvar"
- multi-line strings: """

great

for

shaders"""

- for-each loops for lists and maps
- dynamic method handling, .., =>

# more things

- Runs server side as well ( like Node.js )
- Pub - packet and dependency mngmnt ( npm )
- Ofc: WebSocket, WebAudio and WebGL
- Typed Arrays => SIMD
- Operator overwriting !!! ( Just imagine Vector math )
- asserts
- const and final
- covariant Generics
  - Map<String, Node> mynodes = new Map<String, Node>();
- Event Streams ( http://stackoverflow.com/q/16005805/756233 )

# Concurrency

isolates: Modern web browsers, even on mobile platforms, run on multi-core CPUs. To take advantage of all those cores, developers traditionally use shared-memory threads running concurrently. However, shared-state concurrency is error prone and can lead to complicated code. Instead of threads, all Dart code runs inside of isolates. Each isolate has its own memory heap, ensuring that no isolate's state is accessible from any other isolate.

TL;DR=> no shared memory, only msg passing

( http://en.wikipedia.org/wiki/Actor_model )

- GC per Isolate, RPC (theory)
- Futures ( Promises )

# Dart Editor

- auto completion
- code navigation ( ctrl-click )
- debug on exception/error
- step through code
- edit-and-continue debugging
- refactoring
- quick fixes
- static analysis
- everything free and open source

# Links

http://www.dartlang.org
https://code.google.com/p/dart
https://www.dartlang.org/slides/2012/06/io12/Dart-A-Modern-Web-Language.pdf
http://blog.sethladd.com/2012/01/for-loops-in-dart-or-fresh-bindings-for.html
http://www.dartlang.org/docs/technical-overview/
https://plus.google.com/+GoogleDevelopers/posts/czhPXadicGG (SIMD)
http://www.infoq.com/presentations/Dart

# Danke