

Training TypeScript

Module – the 'this' keyword



Peter Kassenaar

info@kassenaar.com

TypeScript Repo

The screenshot shows the GitHub interface for the repository 'PeterKassenaar / voorbeeldenTypeScript'. The left sidebar displays the file tree with a 'src' directory containing various TypeScript files (e.g., 00-this-primer.ts, 01-strong-typing.ts, etc.). The main content area shows the repository details, including the name, description ('Voorbeelden en demo's bij de training TypeScript (Peter Kassenaar)'), and statistics (9 commits, 1 branch, 0 packages, 0 releases, 1 contributor). Below this, a table lists the repository's files and their commit history. The 'README.md' file is highlighted, showing its content: 'voorbeeldenTypeScript' and 'Voorbeelden en demo's bij de training TypeScript (Peter Kassenaar)'. The footer of the page includes the GitHub logo, copyright information (© 2019 GitHub, Inc.), and links to Terms, Privacy, Security, Status, Help, Contact GitHub, Pricing, API, Training, Blog, and About.

PeterKassenaar / voorbeeldenTypeScript

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

Voorbeelden en demo's bij de training TypeScript (Peter Kassenaar) Edit

Manage topics

9 commits 1 branch 0 packages 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

File	Commit	Time
src	Updated Type Query example	2 years ago
.gitignore	Eerste versie oefeningen en voorbeeldcode	4 years ago
README.md	Initial commit	4 years ago
gulpfile.js	Eerste versie oefeningen en voorbeeldcode	4 years ago
package.json	Eerste versie oefeningen en voorbeeldcode	4 years ago

README.md

voorbeeldenTypeScript

Voorbeelden en demo's bij de training TypeScript (Peter Kassenaar)

© 2019 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About

<https://github.com/PeterKassenaar/voorbeeldenTypeScript>

Both ES6 and TypeScript know `this` keyword



But what exactly is `this`?

- One of the design flaws in JavaScript (20+ yrs ago)
- `this` has a different meaning, depending on the context
 - TypeScript flattens this behavior by providing a generic context
 - 'Lexical `this`'
 - Only inside classes!
- The question is: "who is calling `this`?"
 - "the value of `this` is determined by how a function is called"

1. Invoking via function

```
//1. Via function
function myFunction() {
    console.log('Function :::', this);
}
myFunction();

// The window is invoking the function, so 'this'
points to Window object.
```

2. Via Object


```
// 2. Via Object
const myObj = {
  myMethod() {
    console.log('Object:::', this);
  }
};
myObj.myMethod();

// The object myObj is invoking the function, so
// 'this' points to the object
```


3. Via TypeScript class


```
class MyClass {  
    name: string = 'Peter';  
    myMethod() {  
        console.log('Class:::', this);  
    }  
}  
  
const myClass = new MyClass();  
myClass.myMethod();  
  
// The class is invoking the function, so 'this'  
points to the class.
```

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/this>


MDN web docs
moz://a

Technologies ▾References & Guides ▾Feedback ▾

Sign in 



this

Languages [Edit](#) 

Jump to: [Syntax](#) [Global Context](#) [Function context](#) [Specifications](#) [Browser compatibility](#) [See also](#)

[Web technology for developers](#) ▸
[JavaScript](#) ▸ [JavaScript reference](#) ▸
[Expressions and operators](#) ▸ [this](#)

Related Topics

JavaScript

Tutorials:

- ▶ Complete beginners
- ▶ JavaScript Guide
- ▶ Intermediate
- ▶ Advanced

References:

- ▶ Built-in objects
- ▼ Expressions & operators
 - Arithmetic operators
 - ⚠ Array comprehensions
 - Assignment operators
 - Bitwise operators
 - Comma operator
 - Comparison operators

 **JavaScript Demo: Expressions - this**

Workshop

- Create a TypeScript page with a plain old JavaScript function
- Compile it, run inside NodeJS or in HTML-page
- What does `this` point at?
- Create a function inside that function. For example a `setTimeout()` with a callback function.
- Do a `console.log()` inside the callback. What does `this` point at?
- Create a TypeScript class and copy the `setTimeout` function to the class. Where does `this` point at now?
- Optional: use the explicitly typed `this` inside a DOM-function.

See example [00-this-primer/00-this-primer.ts](#)

