



Turbo powered Web Apps





Am Anfang war HTML.




WELCOME TO THE NINETIES!

Remember the nineties? The days when *nobody* used CSS, [Mosaic](#) was still a thing and [HTML3.2](#) tags were written in ALL CAPS because they were IMPORTANT goddammit.

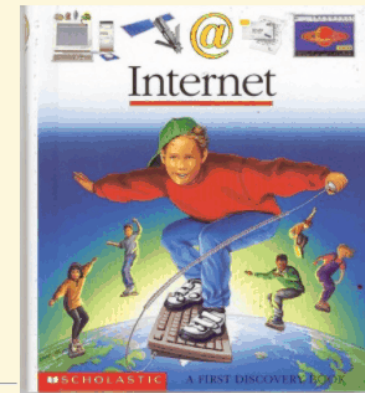
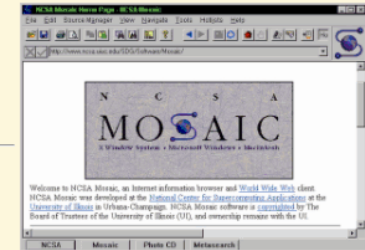
It was a **golden** age when **anything** seemed possible and there was no Facebook to monetize our dreams.

My name is [Charlie Harvey](#) and I miss the websites of the nineties. Websites that were made by real people not fancy pants designers or people who knew what they were doing. You could fit everything on one lovingly crafted HTML page with all your favourite GIFS. We were exploring the cyberspace together!

- Do you miss the nineties too?
- Have you got a nineties site design you would like featured [HERE](#)?

Send me  and let's bring the nineties web back to life!

I will **HYPERLINK** all the best designs from this website!



NEED INSPIRATION?

If you are new to the **WORLD WIDE WEB**, click the interactive **HYPERLINK** to watch *The Kids Guide To The Internet* and learn about "all that



HTML 3.2 Reference Specification

W3C Recommendation 14-Jan-1997

Superseded 15-March-2018

Author: [Dave Raggett](mailto:dsc@w3.org) <dsc@w3.org>

Status of this document

This document has been reviewed by W3C members and other interested parties and has been endorsed by the Director as a W3C Recommendation.

This specification is a [Superseded Recommendation](#). A newer specification exists that is recommended for new adoption in place of this specification. New implementations should follow the [latest version](#) of the HTML specification.

A list of current W3C Recommendations and other technical documents can be found at <https://www.w3.org/TR/>.

Abstract

The HyperText Markup Language (HTML) is a simple markup language used to create hypertext documents that are portable from one platform to another. HTML documents are SGML documents with generic semantics that are appropriate for representing information from a wide range of applications. This specification defines HTML version 3.2. HTML 3.2 aims to capture recommended practice as of early '96 and as such to be used as a replacement for HTML 2.0 ([REC 1866](#)).

Contents

- [Introduction to HTML 3.2](#)
- [HTML as an SGML application](#)
- [The Structure of HTML documents](#)
- [The HEAD element and its children](#)
- [The BODY element and its children](#)
- [Sample SGML Open Catalog for HTML 3.2](#)
- [SGML Declaration for HTML 3.2](#)
- [HTML 3.2 Document Type Definition](#)
- [Character Entities for ISO Latin-1](#)
- [Table of printable Latin-1 Character codes](#)
- [Acknowledgements](#)
- [Further Reading ...](#)

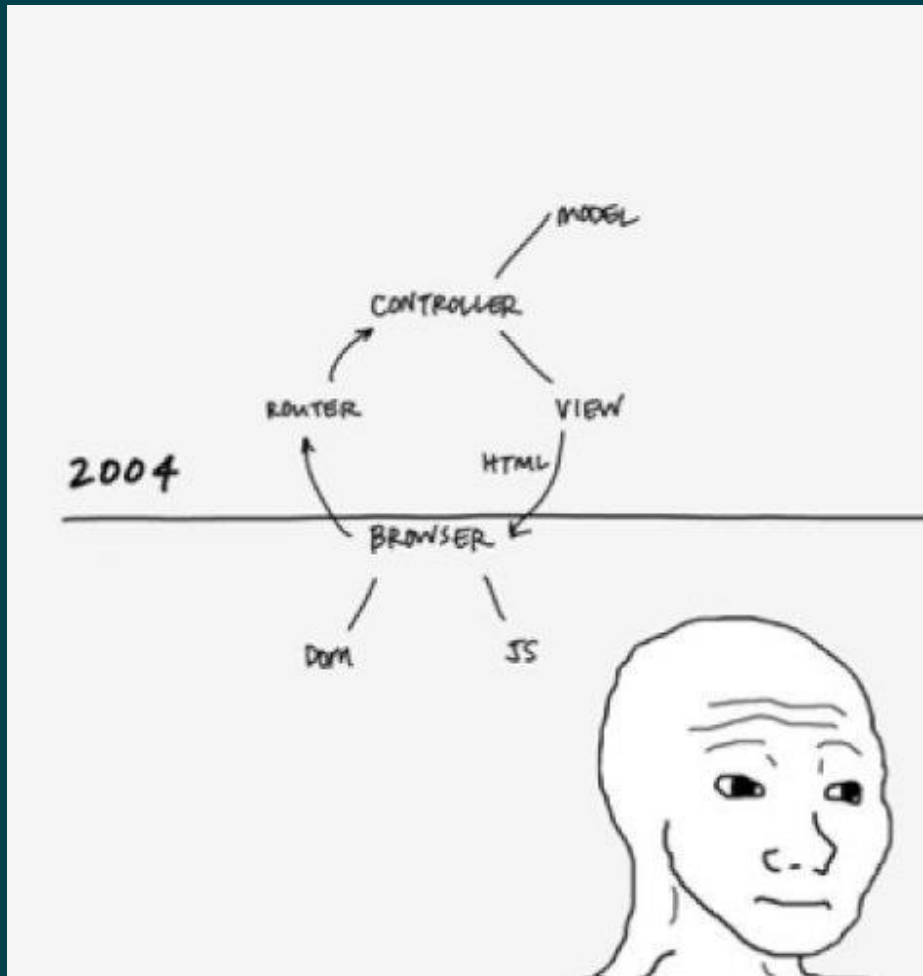
Introduction to HTML 3.2

HTML 3.2 is W3C's specification for HTML, developed in early '96 together with vendors including IBM, Microsoft, Netscape Communications Corporation, Novell, SoftQuad, Spyglass, and Sun Microsystems. HTML 3.2 adds widely deployed features such as tables, applets and text flow around images, while providing full backwards compatibility with the existing standard HTML 2.0.

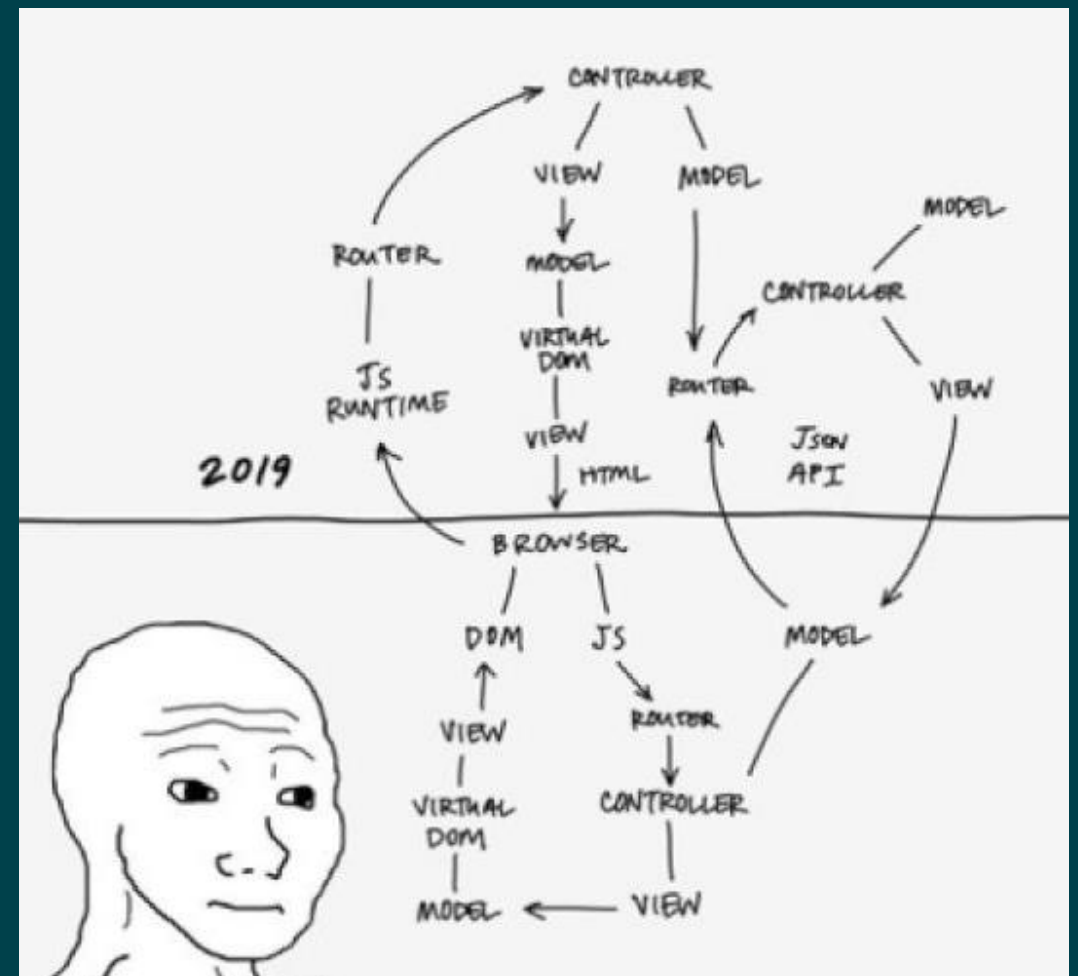
W3C is continuing to work with vendors on extensions for accessibility features, multimedia objects, scripting, style sheets, layout, forms, math and internationalization. W3C plans on incorporating this work in further versions of HTML.

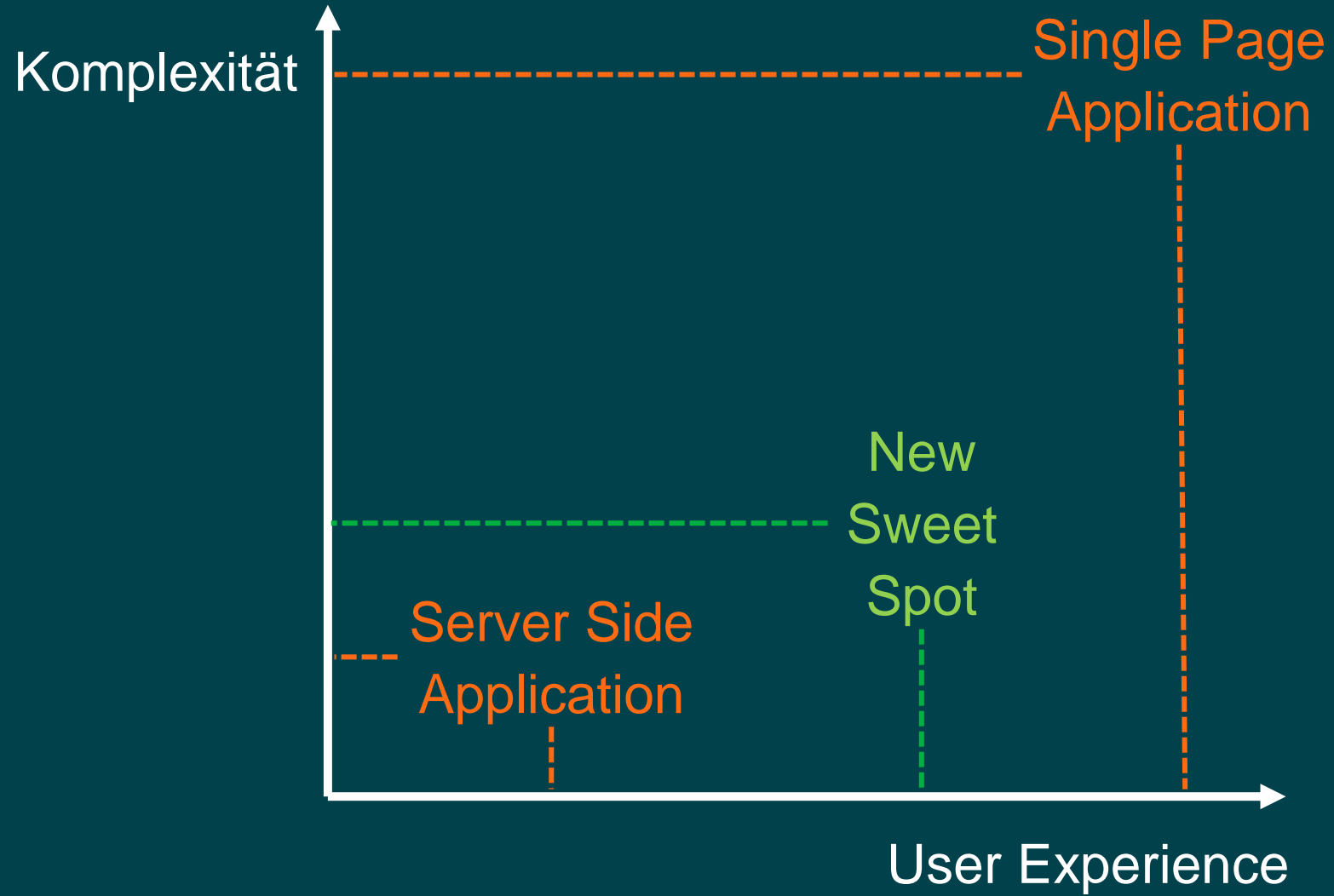


How it started:



How it's going:

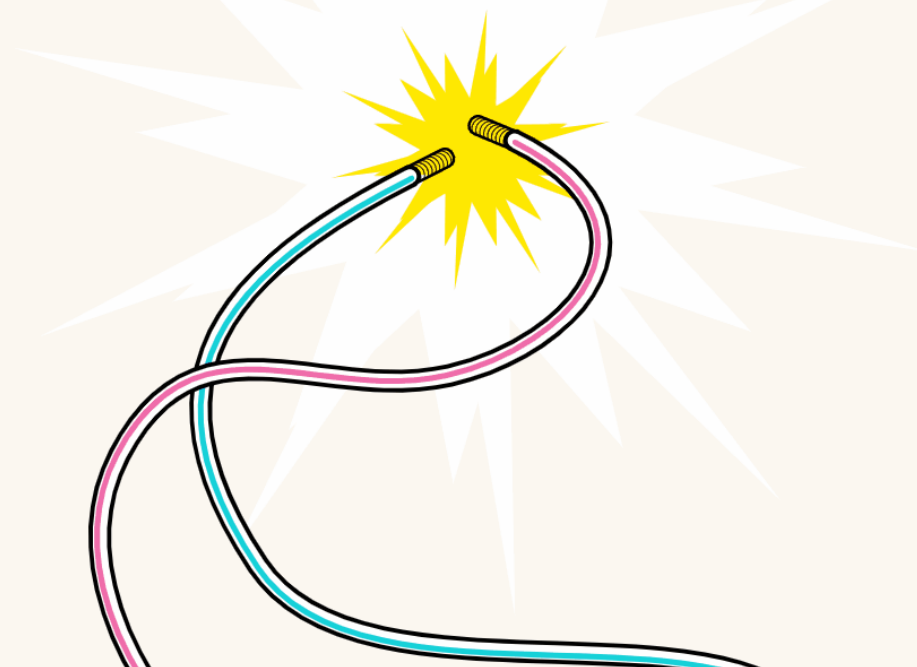






HOTWIRE

HTML OVER THE WIRE

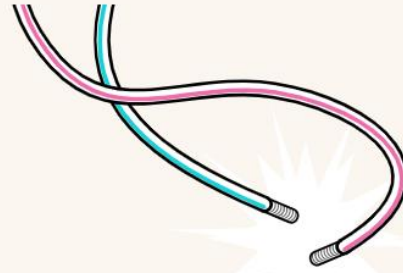


HOTWIRE



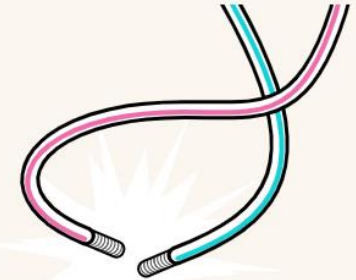
 **TURBO**

- # Beschleunigt Seitenwechsel und Formulare
- # Aufteilung von Seiten in Komponenten
- # Dynamische Updates



 **STIMULUS**

- # bescheidenes Java Script Framework für Controller



 **STRADA**

- # hybride mobile Applications mit nativem Code und Web Views



Navigation mit Turbo Drive



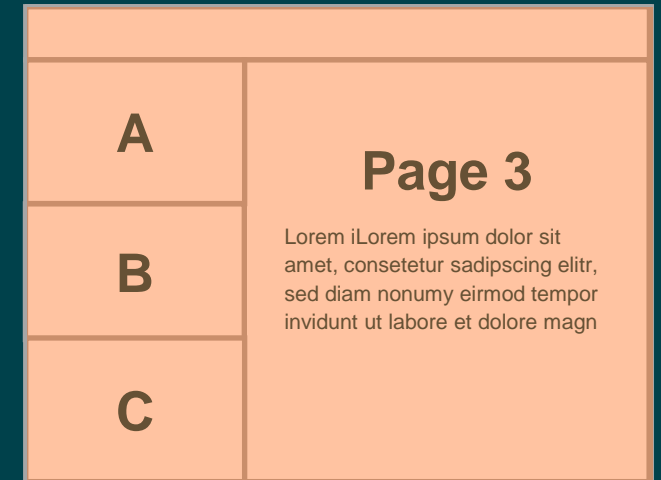
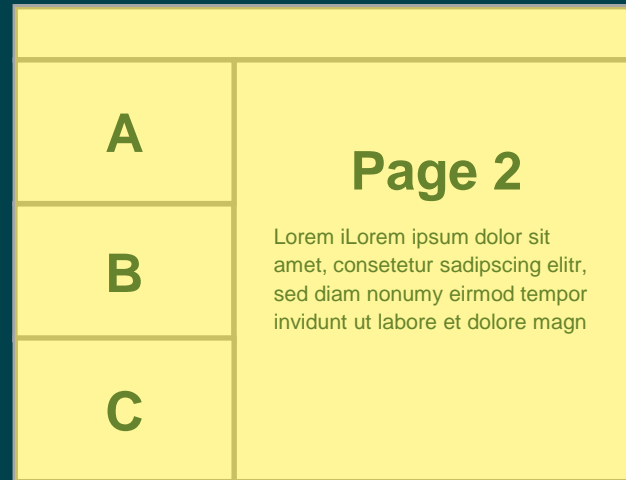
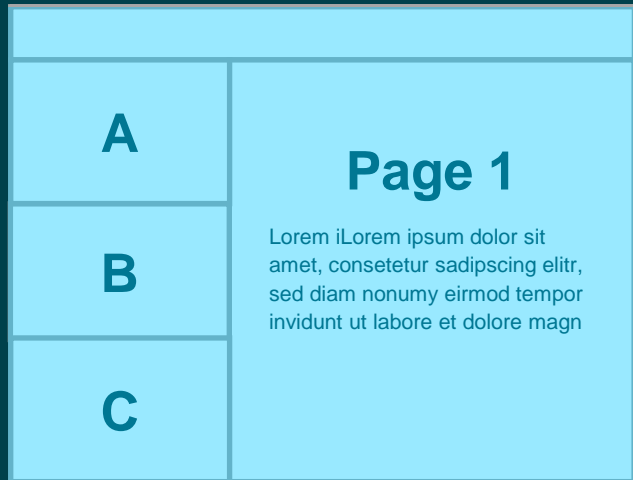
Navigation in klassischer Server Side Application



1. Scroll Position wird zurückgesetzt
2. ungespeicherte Änderungen gehen verloren



1. Scroll Position wird zurückgesetzt
2. ungespeicherte Änderungen gehen verloren



1. HTML rendern
2. CSS (+ Schriften) verarbeiten
3. JavaScript verarbeiten und JavaScript VM starten



1. HTML rendern
2. CSS (+ Schriften) verarbeiten
3. JavaScript verarbeiten und JavaScript VM starten



1. HTML rendern
2. CSS (+ Schriften) verarbeiten
3. JavaScript verarbeiten und JavaScript VM starten





Turbo einlegen

```
<head>  
  <script src="//@hotwired/turbo@7.0.0-beta.5/turbo.es5-umd.js"/>  
</head>
```

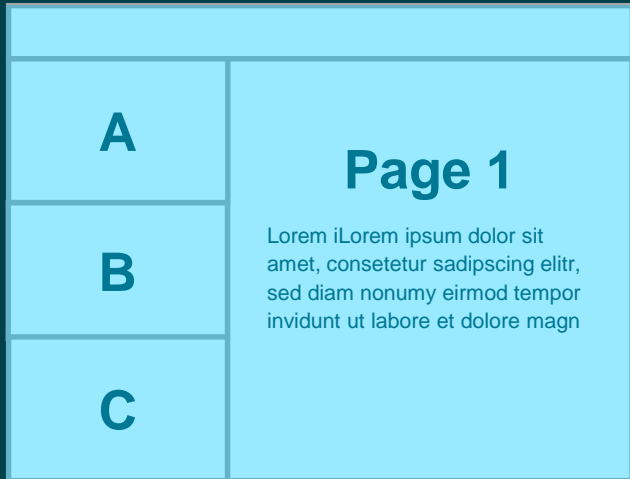
Navigation Server Side Application + Turbo



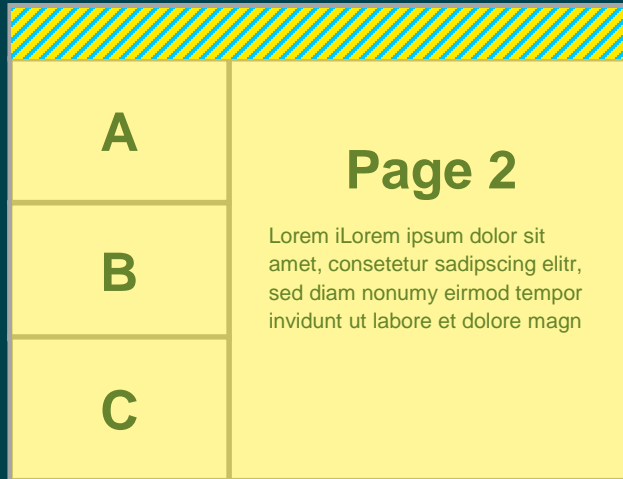
1. ungespeicherte Änderungen gehen verloren



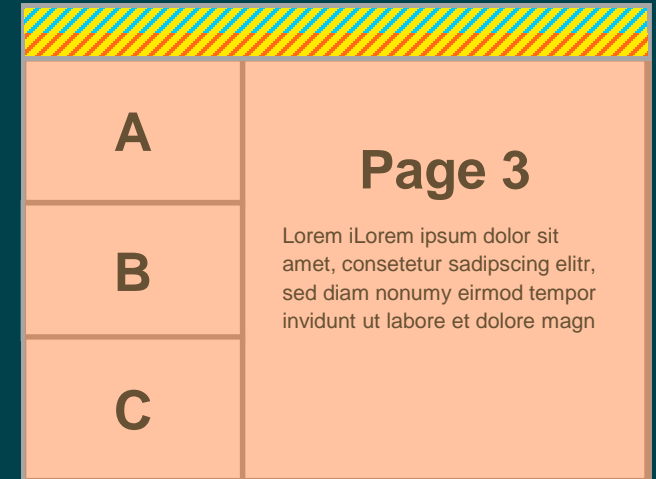
1. ungespeicherte Änderungen gehen verloren



Seite aus Cache holen



Seite aus Cache holen



1. HTML rendern
2. CSS (+ Schriften) verarbeiten
3. JavaScript verarbeiten und JavaScript VM starten



1. CSS und JavaScript mergen und verarbeiten
2. HTML Body austauschen



1. CSS und JavaScript mergen und verarbeiten
2. HTML Body austauschen





DEMO Turbo Navigation

http://localhost:3000/samples/turbo-drive_01/page1.html





Navigation mit Turbo Drive

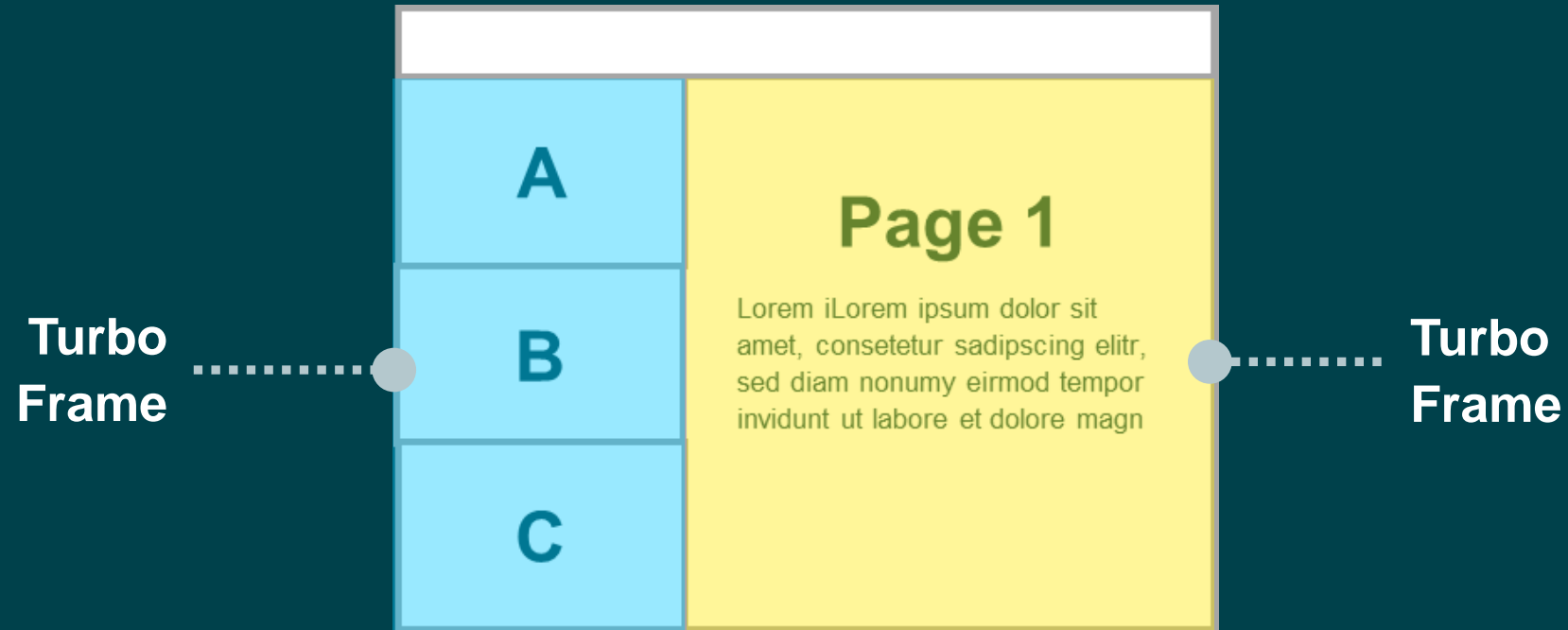
- # Turbo übernimmt alle Links und Formulare aus gleichem Origin (auch Redirects)
- # HTML Head wird gemergt, Body ersetzt
- # Während Seite lädt wird Version aus Cache angezeigt



Seiten zerlegen mit Turbo Frames



Von Seiten zu Turbo Frames



- # Seite wird in Turbo Frames zerlegt die einzeln aktualisiert werden
- # Links und Formular Eingaben beziehen sich nur auf Frame



Turbo Frame Request



```
<html>
  <nav></nav>
  <turbo-frame id="my_frame">
    <h1>My Frame Content</h1>
    <a href="result.html">A framed link</a>
  </turbo-frame>
</html>
```

```
<html>
  <nav></nav>
  <turbo-frame id="my_frame">
    <h1>My Result Content</h1>
  </turbo-frame>
</html>
```

GET result.html
Turbo-Frame: my-frame

```
<html>
  <nav></nav>
  <turbo-frame id="my_frame">
    <h1>My Result Content</h1>
  </turbo-frame>
  <div>more content ...</div>
</html>
```



Turbo Frame Request



```
<html>
  <nav></nav>
  <turbo-frame id="my_frame">
    <h1>My Frame Content</h1>
    <a href="result.html">A framed link</a>
  </turbo-frame>
</html>
```

```
<html>
  <nav></nav>
  <turbo-frame id="my_frame">
    <h1>My Result Content</h1>
  </turbo-frame>
</html>
```

GET result.html
Turbo-Frame: my-frame

```
<turbo-frame id="my_frame">
  <h1>My Result Content</h1>
</turbo-frame>
```



Aus Turbo Frame ausbrechen



```
<a href="result.html" data-turbo-frame="my_frame">Link to Frame</a>
```

```
<turbo-frame id="my_frame"> ← aktualisiert
```

```
  <form action="/submit" method="GET" >
```

```
    <input type="text">My Input</input>
```

```
    <button type="submit">Submit</button>
```

```
  </form>
```

```
</turbo-frame>
```

Formular in Turbo Frame

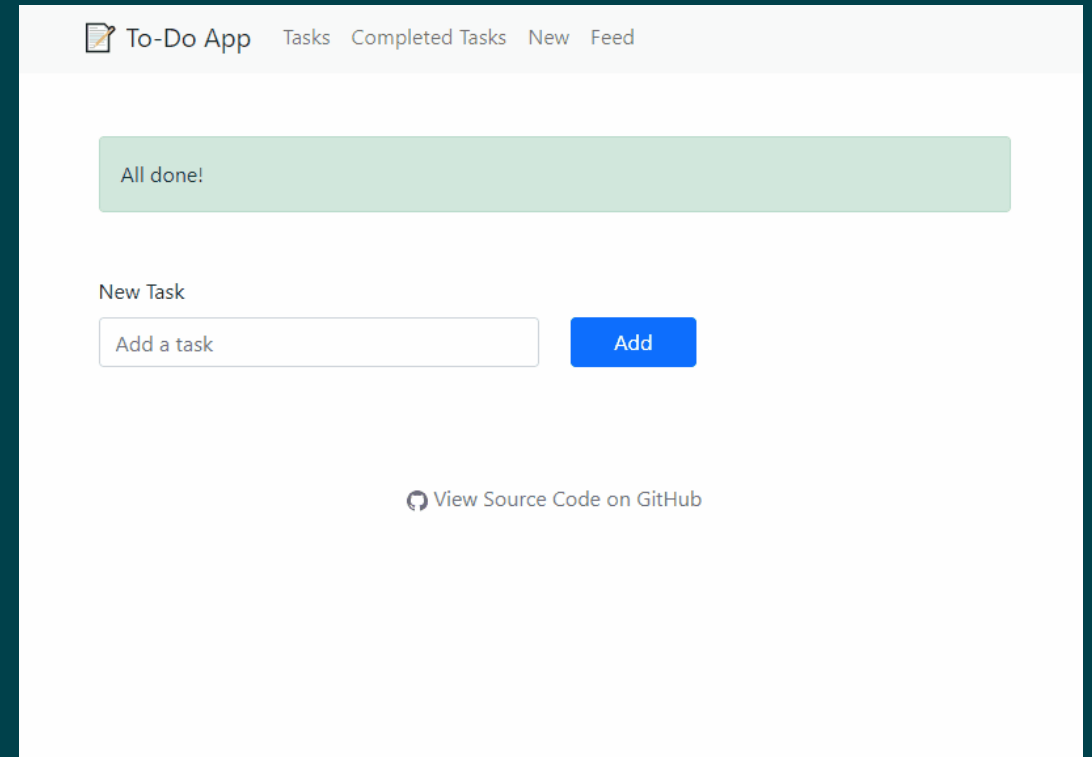


```
<turbo-frame id="my_frame">  
  <form action="/submit" method="GET" >  
    <input type="text">My Input</input>  
    <button type="submit">Submit</button>  
  </form>  
</turbo-frame>
```


Lazy Turbo Frame



```
<turbo-frame id="my_frame" src="feed-frame">  
  <div class="spinner-border" role="status">  
    <span class="sr-only">Loading...</span>  
  </div>  
</turbo-frame>
```





DEMO Turbo Frames

http://localhost:3000/samples/turbo-frame_01/index.html





Seite zerlegen mit Turbo Frames

- # Seite wird in Turbo Frames zerlegt die einzeln aktualisiert werden
- # Links und Formulare wirken nur innerhalb des Frames
- # Frames können lazy geladen werden



Dynamische Updates mit Turbo Streams



Formular mit Turbo Stream



```
<html>
  <nav></nav>
  <form action="/submit"
        method="POST"
        id="my_form">
    <input type="text">My Input</input>
    <button type="submit">Submit</button>
  </form>
</html>
```

POST result.html

Accept: text/vnd.turbo-stream.html

```
<turbo-stream action="replace" target="my_form">
  <template>
    <p>Thanks for your Input!</p>
  </template>
</turbo-stream>
```



```
<html>
  <nav></nav>
  <p>Thanks for your Input!</p>
</html>
```



Formular mit Turbo Stream



```
<turbo-stream action="replace" target="my_form">
  <template>
    <p>Thanks for your Input!</p>
  </template>
</turbo-stream>
```

Turbo Streams erlauben Aktionen

- # append
- # prepend
- # replace
- # update
- # remove



Server Sent Events (SSE) per Turbo Stream



```
<script>
Turbo.connectStreamSource(
  new EventSource("/events?stream=messages")
);
</script>

<div id="feed-frame">
  <div class="task">Task 0</div>
</div>
```



```
<div id="feed-frame">
  <div class="task">Task 1</div>
  <div class="task">Task 0</div>
</div>
```

EventStream Id: 1



```
<turbo-stream action="prepend" target="feed-frame">
  <template>
    <div class="task">Task 1</div>
  </template>
</turbo-stream>
```

Server Sent Events (SSE) per Web Socket



```
<script>
Turbo.connectStreamSource(
  new WebSocket("ws://localhost:3000/ws")
);
</script>

<div target="feed-frame">
  <div class="task">Task 0</div>
</div>
```



```
<div target="feed-frame">
  <div class="task">Task 1</div>
  <div class="task">Task 0</div>
</div>
```

Push Data



```
<turbo-stream action="prepend" target="feed-frame">
  <template>
    <div class="task">Task 1</div>
  </template>
</turbo-stream>
```




DEMO Turbo Streams

<http://localhost:3000/task/new>





Dynamische Updates mit Turbo Streams

- # mit Streams können feingranular Teile der Seite aktualisiert werden
- # eigener Turbo Accept Header gibt an ob Client Turbo Streams akzeptiert
- # Live Updates per Server Sent Events oder Web Sockets



Beispiel To-Do App



To-Do App Tasks Completed Tasks New Feed (lazy Frame) Feed (slow) Feed (with SSE) Feed (with WebSocket)

- Wash Dishes ✓
- Make Laundry ✓
- Clean Kitchen ✓

New Task

Add a task

[View Source Code on GitHub](#)

**Turbo Drive
Navigation**

**Formular mit
Turbo Stream**

Turbo Frame



DEMO
To-Do App
<http://localhost:3000/>





HOTWIRE



TURBO

Turbo Drive

- # SPA-Style Navigation
- # Beschleunigt Seitenwechsel und Formulare

Turbo Frames

- # Seite wird in Turbo Frames zerlegt die einzeln aktualisiert werden
- # Links und Formulare wirken innerhalb des Frames

Turbo Streams

- # Feingranulare Updates von Teilen der Seiten
- # Live Updates per Server Sent Events oder WebSockets



Jan Stamer

Solution Architect

jan.stamer@comdirect.de





COMMERZBANK