

Automated Repair of Responsive Web Page Layouts

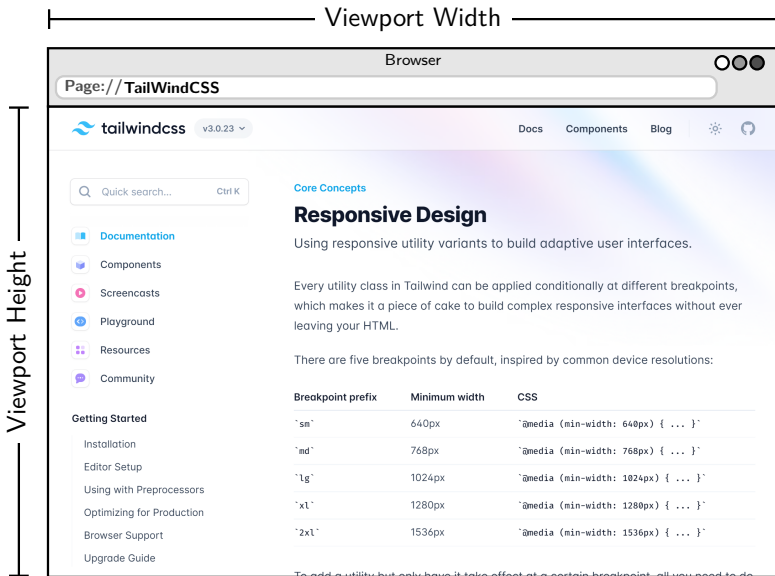
Ibrahim Althomali ¹ Gregory Kapfhammer ² Phil McMinn ¹

¹University of Sheffield

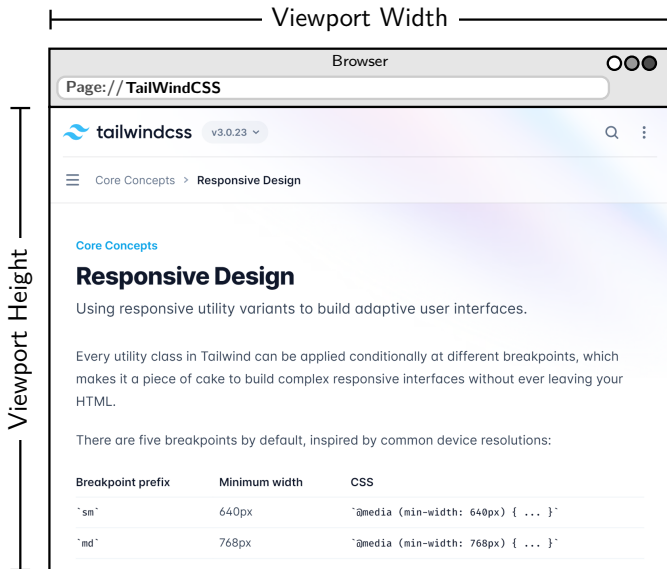
²Allegheny College

March 19, 2022

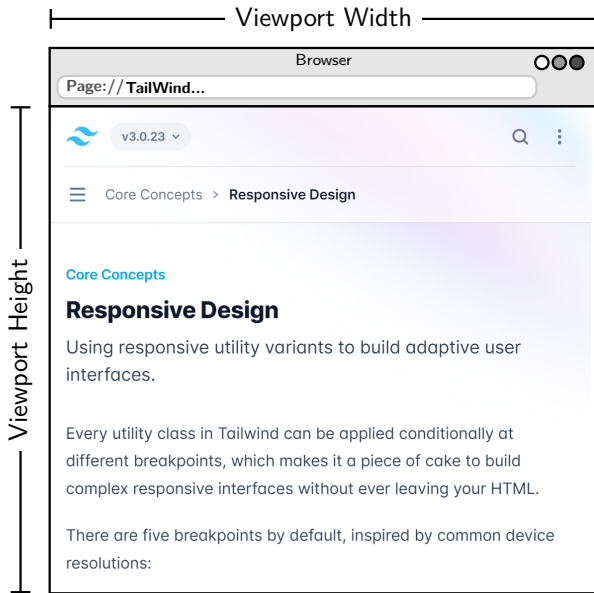
Responsive Web Design



Responsive Web Design



Responsive Web Design

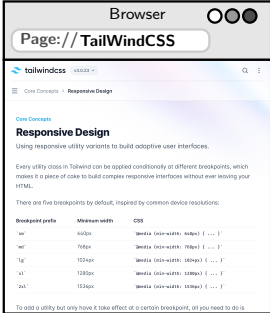


Responsive Web Design

Mobile

The mobile view of the TailwindCSS documentation page. The browser address bar shows 'Page://TailWind...'. The page content is scaled down to fit the mobile screen, with the 'Responsive Design' section being the primary focus.

Tablet

The tablet view of the TailwindCSS documentation page. The browser address bar shows 'Page://TailWindCSS'. The page content is scaled up to fit the tablet screen, with the 'Responsive Design' section being the primary focus.

Desktop

The desktop view of the TailwindCSS documentation page. The browser address bar shows 'Page://TailWindCSS'. The page content is scaled up to fit the desktop screen, with the 'Responsive Design' section being the primary focus. A table of breakpoints is visible in the lower right.

Browser

Page://TailWindCSS

tailwindcss v3.0.23

Core Concepts > Responsive Design

Core Concepts

Responsive Design

Using responsive utility variants to build adaptive user interfaces.

Every utility class in Tailwind can be applied conditionally at different breakpoints, which makes it a piece of cake to build complex responsive interfaces without ever leaving your HTML.

There are five breakpoints by default, inspired by common device resolutions:

Breakpoint prefix	Minimum width	CSS
'sm'	640px	'media (<min-width: 640px) { ... }'
'md'	768px	'media (<min-width: 768px) { ... }'
'lg'	1024px	'media (<min-width: 1024px) { ... }'
'xl'	1280px	'media (<min-width: 1280px) { ... }'
'2xl'	1536px	'media (<min-width: 1536px) { ... }'

To add a utility but only have it take effect at a certain breakpoint, all you need to do is

Browser

Page://TailWindCSS

tailwindcss v3.0.23

Docs Components Blog

Quick search...

Core Concepts

Documentation

Components

Screenshots

Playground

Resources

Community

Getting Started

Installation

Editor Setup

Using with Preprocessors

Optimizing for Production

Browser Support

Upgrade Guide

Core Concepts

Utility First Fundamentals

Hover, Focus, and Other States

Responsive Design

Responsive Design

Using responsive utility variants to build adaptive user interfaces.

Every utility class in Tailwind can be applied conditionally at different breakpoints, which makes it a piece of cake to build complex responsive interfaces without ever leaving your HTML.

There are five breakpoints by default, inspired by common device resolutions:

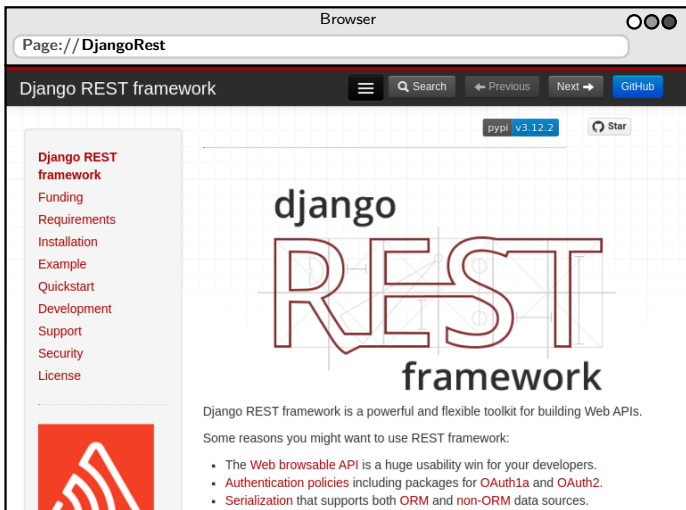
Breakpoint prefix	Minimum width	CSS
'sm'	640px	'media (<min-width: 640px) { ... }'
'md'	768px	'media (<min-width: 768px) { ... }'
'lg'	1024px	'media (<min-width: 1024px) { ... }'
'xl'	1280px	'media (<min-width: 1280px) { ... }'
'2xl'	1536px	'media (<min-width: 1536px) { ... }'

To add a utility but only have it take effect at a certain breakpoint, all you need to do is

prefix the utility with the breakpoint name, followed by the ':' character:

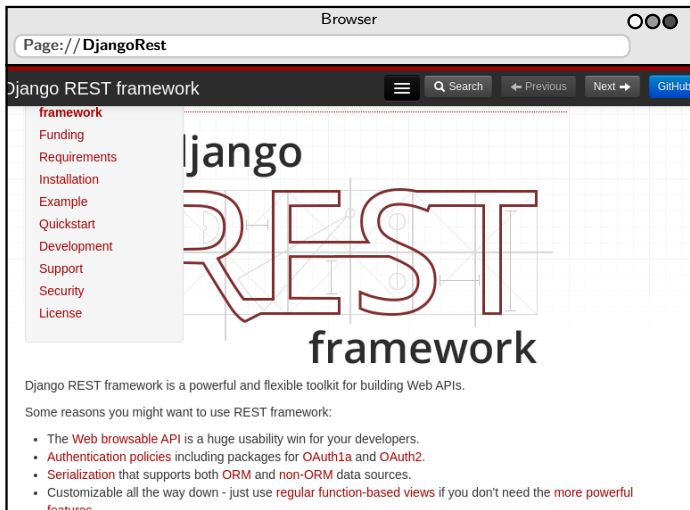
```
<div class="w-16 md:w-32 lg:w-48" ></div>
```

Responsive Layout Failure



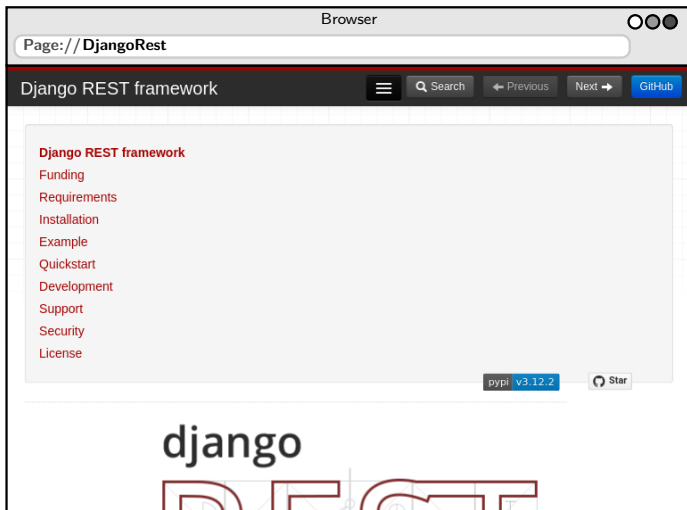
At a wider viewport width (No Failure)

Responsive Layout Failure



A viewport width with the layout failure

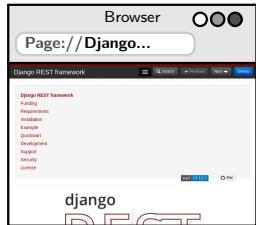
Responsive Layout Failure



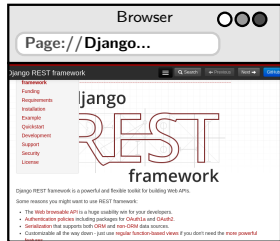
At a narrower viewport width (No Failure)

Testing for Layout Failures

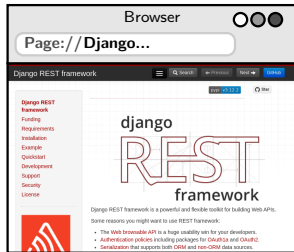
Narrower



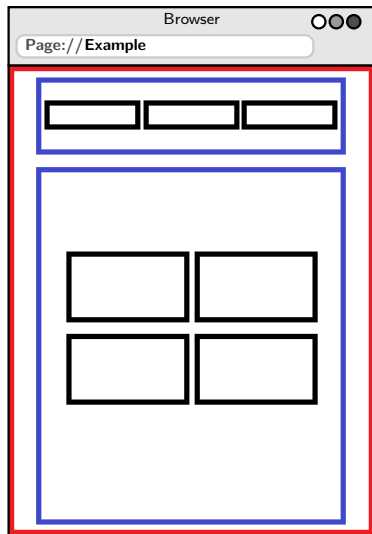
Failure Range



Wider



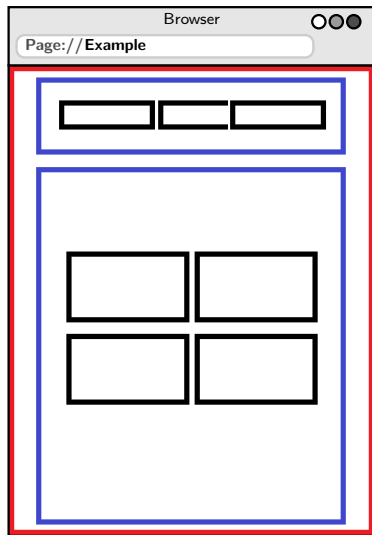
Automated Testing for Layout Failures – ReDeCheck



Responsive Layout Graph

- ▷ Element Collision
- ▷ Element Protrusion
- ▷ Viewport Protrusion
- ▷ Element Wrapping
- ▷ Small-range

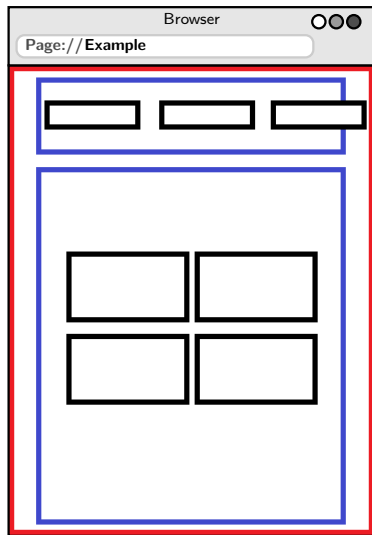
Automated Testing for Layout Failures – ReDeCheck



Responsive Layout Graph

- ▷ Element Collision
- ▷ Element Protrusion
- ▷ Viewport Protrusion
- ▷ Element Wrapping
- ▷ Small-range

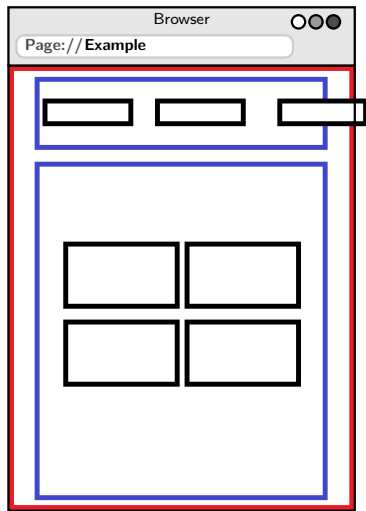
Automated Testing for Layout Failures – ReDeCheck



Responsive Layout Graph

- ▷ Element Collision
- ▷ Element Protrusion
- ▷ Viewport Protrusion
- ▷ Element Wrapping
- ▷ Small-range

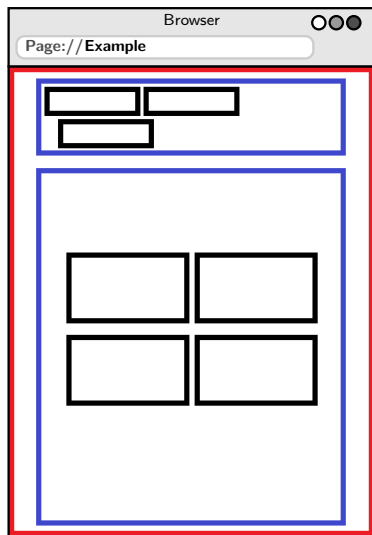
Automated Testing for Layout Failures – ReDeCheck



Responsive Layout Graph

- ▷ Element Collision
- ▷ Element Protrusion
- ▷ Viewport Protrusion
- ▷ Element Wrapping
- ▷ Small-range

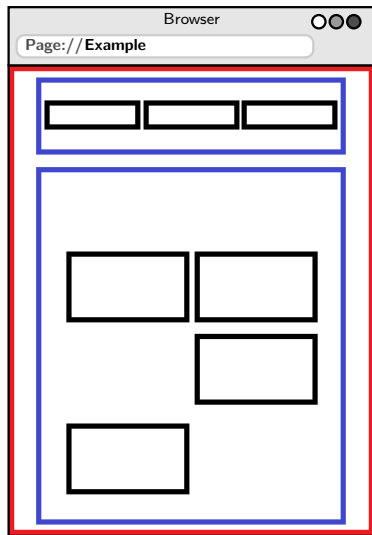
Automated Testing for Layout Failures – ReDeCheck



Responsive Layout Graph

- ▷ Element Collision
- ▷ Element Protrusion
- ▷ Viewport Protrusion
- ▷ Element Wrapping
- ▷ Small-range

Automated Testing for Layout Failures – ReDeCheck

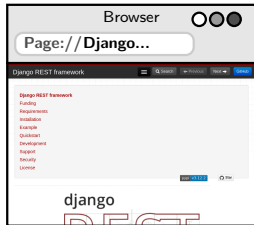


Responsive Layout Graph

- ▷ Element Collision
- ▷ Element Protrusion
- ▷ Viewport Protrusion
- ▷ Element Wrapping
- ▷ Small-range

Repairing a Responsive Layout Failure

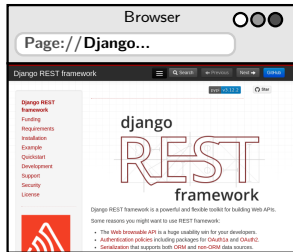
Narrower



Failure Range

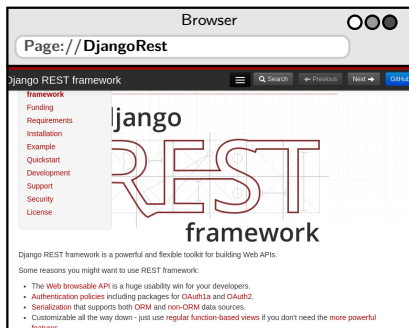


Wider

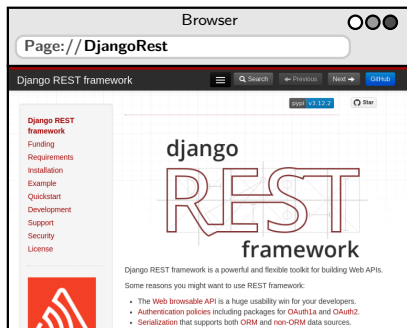


Our Automated Repair Tool – Layout DR

Failure Viewport



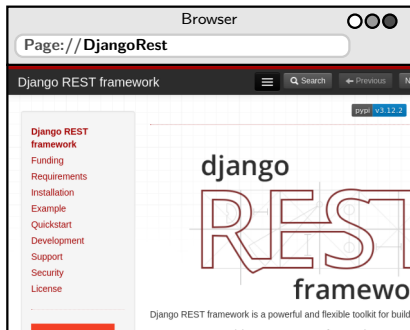
Wider Bordering Viewport



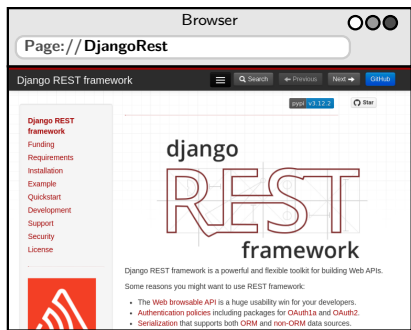
▷ Validation ▷ Harvest CSS ▷ !important ▷ media

Our Automated Repair Tool – Layout DR

Failure Viewport



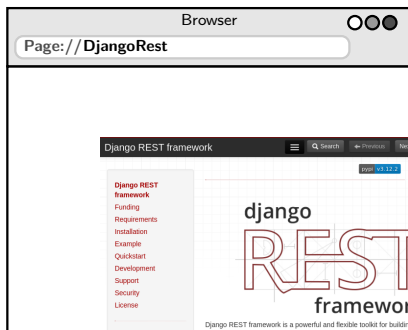
Wider Bordering Viewport



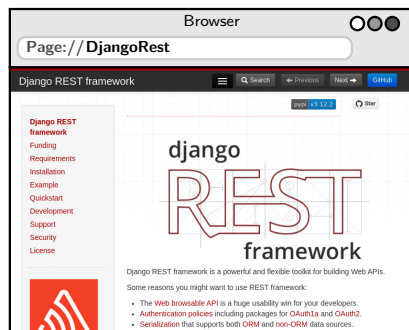
► The harvested CSS is applied to viewport with failure.

Our Automated Repair Tool – Layout DR

Failure Viewport



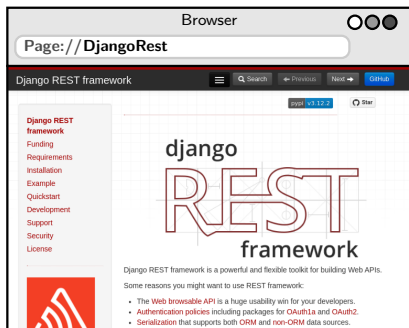
Wider Bordering Viewport



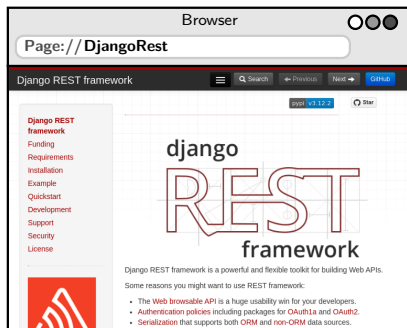
▷ The scale property added to resize the layout.

Our Automated Repair Tool – Layout DR

Failure Viewport



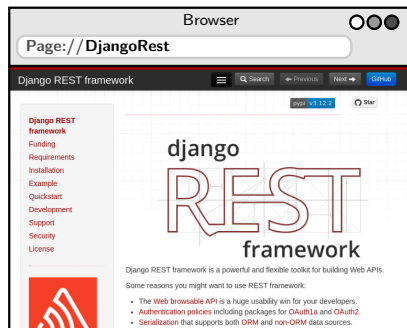
Wider Bordering Viewport



▷ The transform-origin property used to anchor the layout.

Our Automated Repair Tool – Layout DR

— Failure Viewport (Repaired) —



▷ Verify repair using DOM and RLG

Research Objectives

1 – Which of the bordering layouts is likely to result in a repair?

- ▷ Used 55 failures detected from 19 web pages.
- ▷ Tool and I ensured the failure was removed.
- ▷ I checked if the patch is failure-free.

2 – Do humans prefer the repaired page or the original with the failure?

- ▷ Used 20 failures from 14 web pages.
- ▷ Used MTurk for a human study resulting in 738 votes.

3 – How long does it take to generate a patch?

- ▷ Measured the runtime of patch generation phase over 10 runs.

Research Objectives

1 – Which of the bordering layouts is likely to result in a repair?

- ▷ Used 55 failures detected from 19 web pages.
- ▷ Tool and I ensured the failure was removed.
- ▷ I checked if the patch is failure-free.

2 – Do humans prefer the repaired page or the original with the failure?

- ▷ Used 20 failures from 14 web pages.
- ▷ Used MTurk for a human study resulting in 738 votes.

3 – How long does it take to generate a patch?

- ▷ Measured the runtime of patch generation phase over 10 runs.

Research Objectives

1 – Which of the bordering layouts is likely to result in a repair?

- ▷ Used 55 failures detected from 19 web pages.
- ▷ Tool and I ensured the failure was removed.
- ▷ I checked if the patch is failure-free.

2 – Do humans prefer the repaired page or the original with the failure?

- ▷ Used 20 failures from 14 web pages.
- ▷ Used MTurk for a human study resulting in 738 votes.

3 – How long does it take to generate a patch?

- ▷ Measured the runtime of patch generation phase over 10 runs.

Subject Web Pages

Subject	Original URL	Layout Failures	HTML Elements	CSS Declarations
3MinuteJournal	3minutejournal.com	4	80	5499
Ardour	ardour.org	2	222	3774
Bottender	bottender.js.org	5	243	2202
Bower	bower.io	1	370	844
BugMeNot	bugmenot.com	1	42	658
ConsumerReports	consumerreports.org	7	1042	8005
Django	djangoproject.com	1	242	4732
DjangoRest	django-rest-framework.org	1	610	3787
Duolingo	duolingo.com	1	856	4260
ElasticSearch	elastic.co/elasticsearch	2	1243	21467
Honey	joinhoney.com/install	1	461	7903
HotelWiFiTest	hotelwifitest.com	1	359	6746
MantisBT	mantisbt.org	3	247	7731
MarkText	marktext.app	15	560	1890
MidwayMeetup	midwaymeetup.com	1	86	4147
OrchardCore	orchardcore.net	5	234	6352
PepFeed	pepfeed.com	1	343	7276
Selenium	selenium.dev	1	286	4980
WillMyPhoneWork	willmyphonework.net	2	782	6576
Total		55	8308	108829

Research Findings

1 – Which of the bordering layouts is likely to result in a repair?

	Using Narrower Viewport	Using Wider Viewport	Both	Either
Successfully patched	20	55	20	55
Failure-free patches	18	38	16	40

Conclusion – Using the wider viewport for repair is more successful but has higher probability of extending other failure into the patched viewports.

Human Study Page

Browser

Page:// DjangoRest

Description of webpage layout bug:

To the top left-hand side of the page, the letter D in the word "Django" is cut-off.

6) Which webpage do you prefer?

☐ Containing Bug.

☐ Repair A.

☐ Repair B.

Instructions

0% Completed

Submit

⊕ 1/10 ⊕

Browser

Containing Bug

Repair A

Repair B

Django REST framework

framework

Funding

Requirements

Installation

Example


Quickstart

Development

Support

Security

License



The logo for Django REST framework, featuring the word "Django" in a serif font and "REST framework" in a sans-serif font, with a large "R" and "E" in the background.

Django REST framework is a powerful and flexible toolkit for building Web APIs.

Some reasons you might want to use REST framework:

- The **Web browsable API** is a huge usability win for your developers.
- **Authentication policies** including packages for **OAuth1a** and **OAuth2**.
- **Serialization** that supports both **ORM** and **non-ORM** data sources.
- Customizable all the way down - just use **regular function-based views** if you don't need the **more powerful features**.

Human Study Page

Browser

Page:// DjangoRest

Description of webpage layout bug:
To the top left-hand side of the page, the letter D in the word "Django" is cut-off.

6) Which webpage do you prefer?
☐ Containing Bug.
☐ Repair A.
☐ Repair B.

Instructions
0% Completed
Submit
⊕ 1/10 ⊕

Browser

Containing Bug Repair A Repair B

Django REST framework

Django REST framework

Funding

Requirements

Installation

Example

Quickstart

Development

Support

Security

License

python v3.12.2

Star

django

REST

framework

Django REST framework is a powerful and flexible toolkit for building Web APIs.

Some reasons you might want to use REST framework:

- The [Web browsable API](#) is a huge usability win for your developers.
- [Authentication policies](#) including packages for [OAuth1a](#) and [OAuth2](#).
- [Serialization](#) that supports both [ORM](#) and [non-ORM](#) data sources.

Human Study Page

Browser

Page:// DjangoRest

Description of webpage layout bug:

To the top left-hand side of the page, the letter D in the word "Django" is cut-off.

6) Which webpage do you prefer?

☐ Containing Bug.
☐ Repair A.
☐ Repair B.

Instructions

0% Completed

Submit

⊕ 1/10 ⊕

Browser

Containing Bug

Repair A

Repair B

Django REST framework

Django REST framework

Funding

Requirements

Installation

Example

Quickstart

Development

Support

Security

License

py3 v3.12.2

Star

django

REST

Research Findings

2 – Do humans prefer the repaired page or the original with the failure?

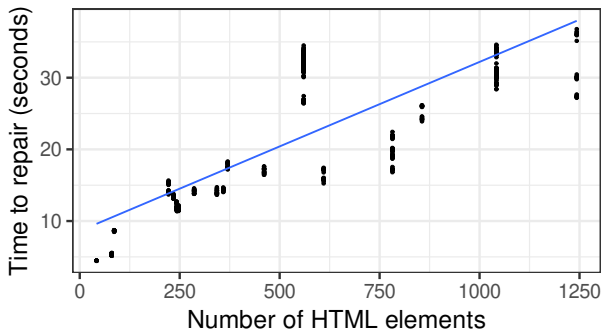
	Original	Narrower Repaired	Wider Repaired	Either Repair
Votes	60 = 8%	292 = 40%	386 = 52%	678 = 92%

Conclusion – A patched web page is preferred 92% of the time over the original with the failure. Moreover, the wider viewport based repairs have a 12% advantage of the narrower.

Research Findings

3 – How long does it take to generate a patch?

▷ **Min:** 4.4 – **Median:** 17.3 – **Mean:** 20.5 – **Max:** 36.8



Conclusion – The runtime of the tool is practical and depends on the number of HTML elements in the web page.

Conclusions and Future Research

Review of findings

- ✓ Successfully repaired all the detected failures.
- ✓ 92% of humans participants preferred the patched subject.
- ✓ Practical runtime averaging 21 seconds per failure.
- ✗ May extend existing failures in the narrower or wider bordering viewport.

Future research

- ▷ Localize the patch to the elements instead of the viewport.