HELLO

Ovidiu Chereches

"LIBRARIE"

BOOK STORE

LIBRARIE

LIBRARY

BIBLIOTECA

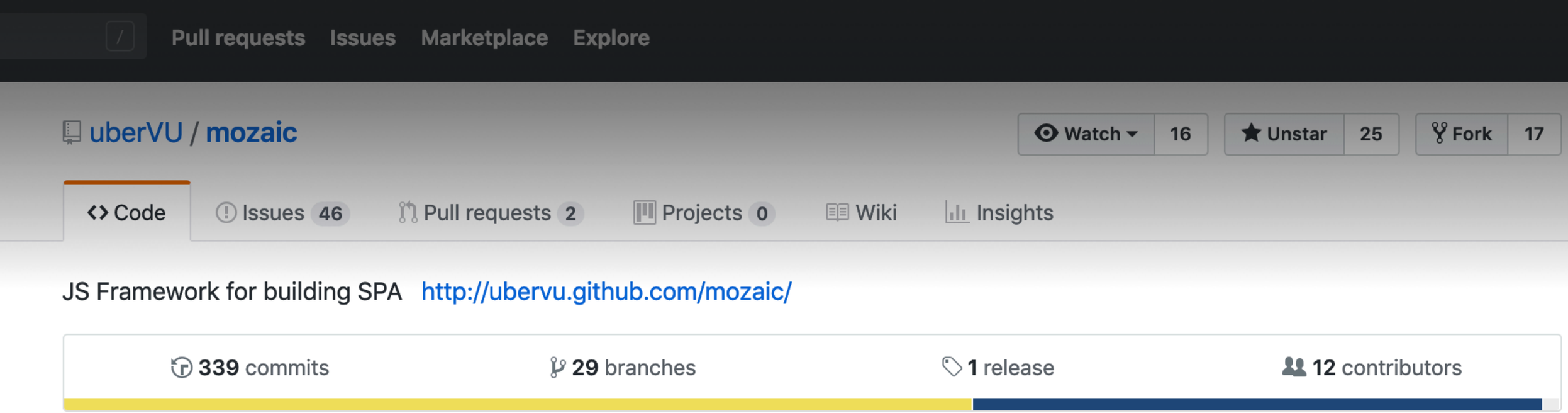
PAST, PRESENT & FUTURE

REACT COSMOS

PAST

MY FIRST "SINGLE PAGE APP"

- RequireJS for dependency management
- Backbone for data management and views
- Handlebars for view templates
- "Mozaic" in-house UI framework



THE "SPA" STRUGGLE

- Unpredictable control flow
- Hard to debug and find source of bugs
- Widget reusability: A false promise

“I HAVE A DREAM”

DEBUG WIDGET VS DEBUG THE APP

“I HAVE A DREAM”

ISOLATED WIDGET ENVIRONMENT

“I HAVE A DREAM”

NAY, ISOLATED COMPONENT ENVIRONMENT

ISOLATED COMPONENT ENV

- Serializable (explicit) component input
- Components as first-class citizens
- An interesting experiment...

WHY ISN'T ALL INPUT SERIALIZABLE IN PRACTICE?

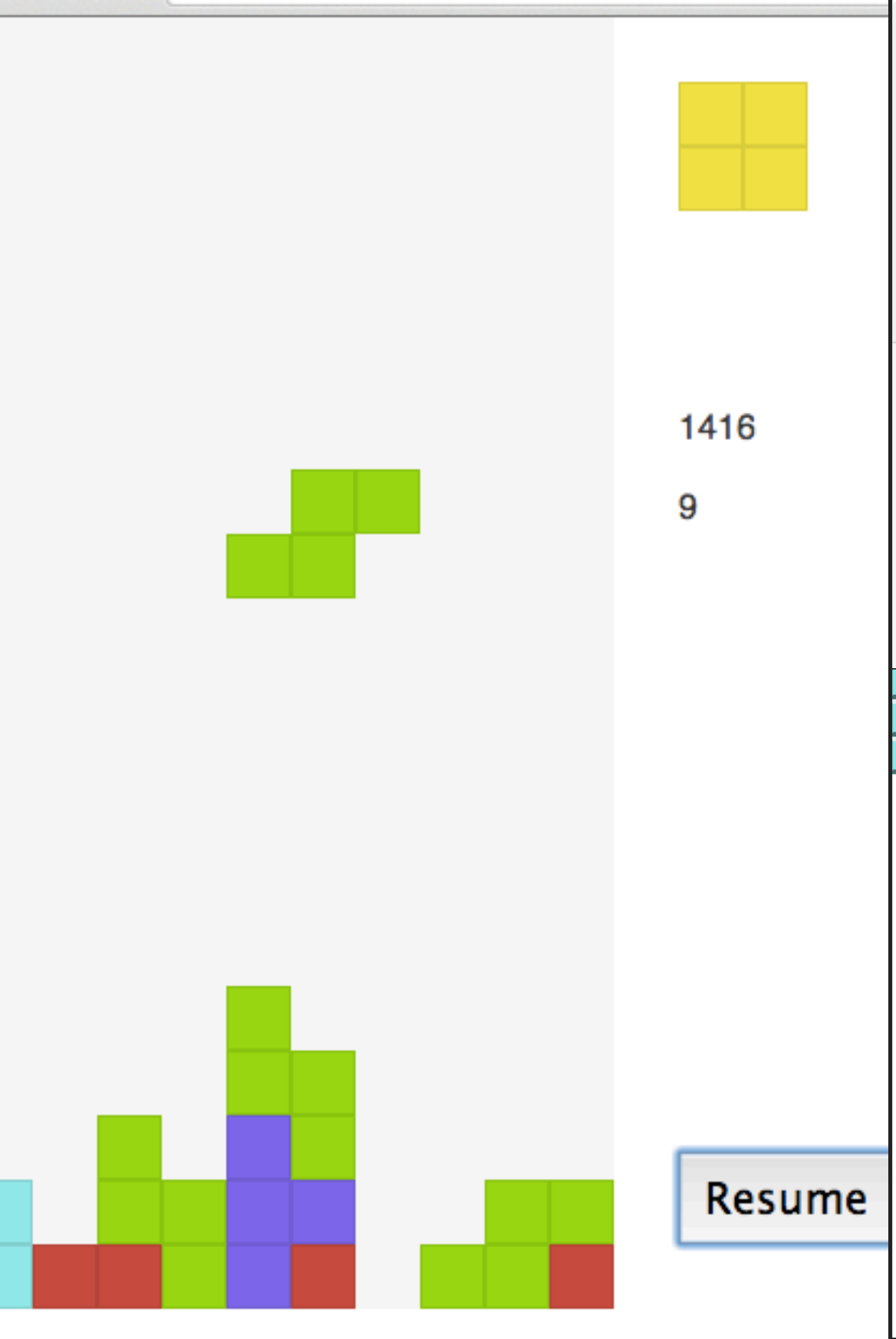
- Input can contain functions (eg. Callback Props in React)
- Some input should be implicit for pragmatic reasons (eg. Context in React)

- add-board-template-wizard/add-board-template-wizard
- add-teammate/base
- add-teammate/filtered
- add-teammate/selected
- add-widget-wizard/base
- analytics/base
- analytics/board-list-with-filter
- analytics/board-list
- analytics/board
- analytics/overview
- autocomplete/autofocus
- autocomplete/base
- autocomplete/no-matches
- autocomplete/no-value
- autocomplete/placeholder
- autocomplete/result-list-visible
- autocomplete/value-with-matches
- board-legend/base

A FIXTURE IS BORN

COSMOS PROTOTYPE

```
export default {
  component: Button,
  props: {
    children: 'Press me will you',
    disabled: true,
    onClick: () => console.log('Yay! Button clicked-ed!')
  }
}
```



Plecare

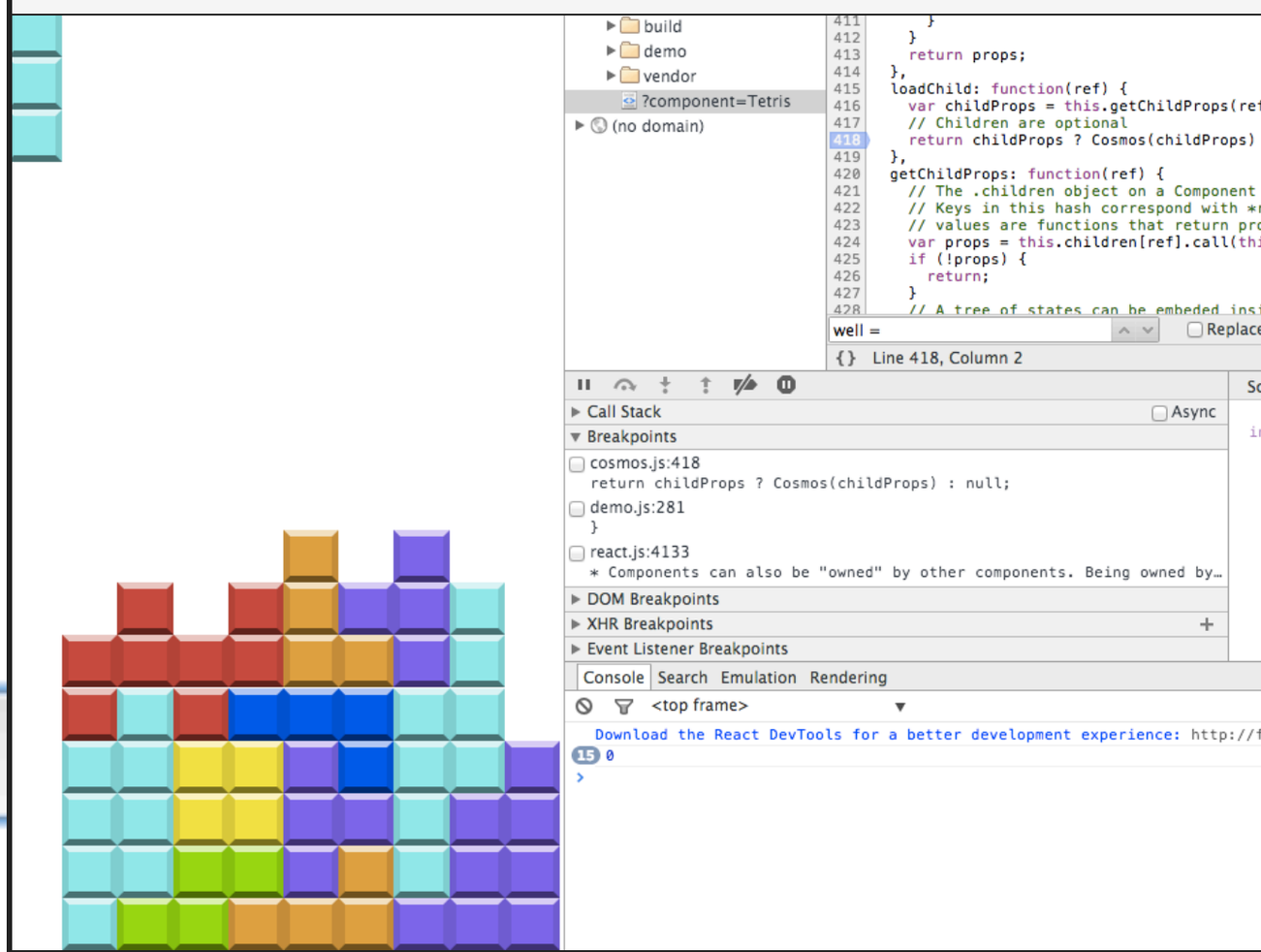
W6 3003 7 iunie 2014

Pleacă din:
Bucureşti H. Coanda (OTP) → Soseşte la: **Londra Luton (LTN)**
11:25 12:50

Întoarcere

W6 3002 15 iunie 2014

Pleacă din:
Londra Luton (LTN) → Soseşte la: **Bucureşti H. Coanda (OTP)**
08:30 13:45



localhost/self/cosmos/

Flatris

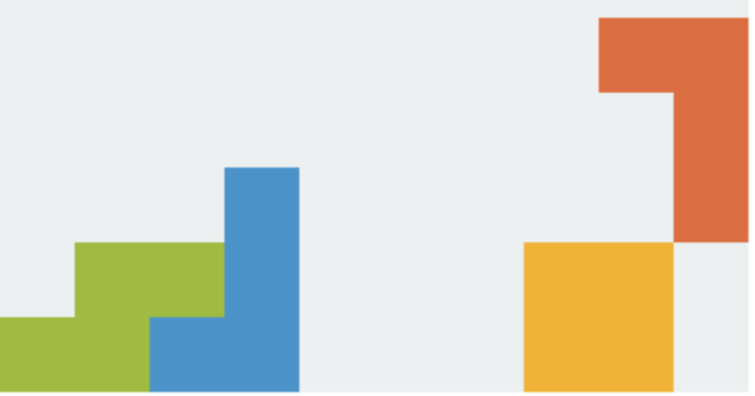
Score

20

Lines Cleared

0

Next Shape



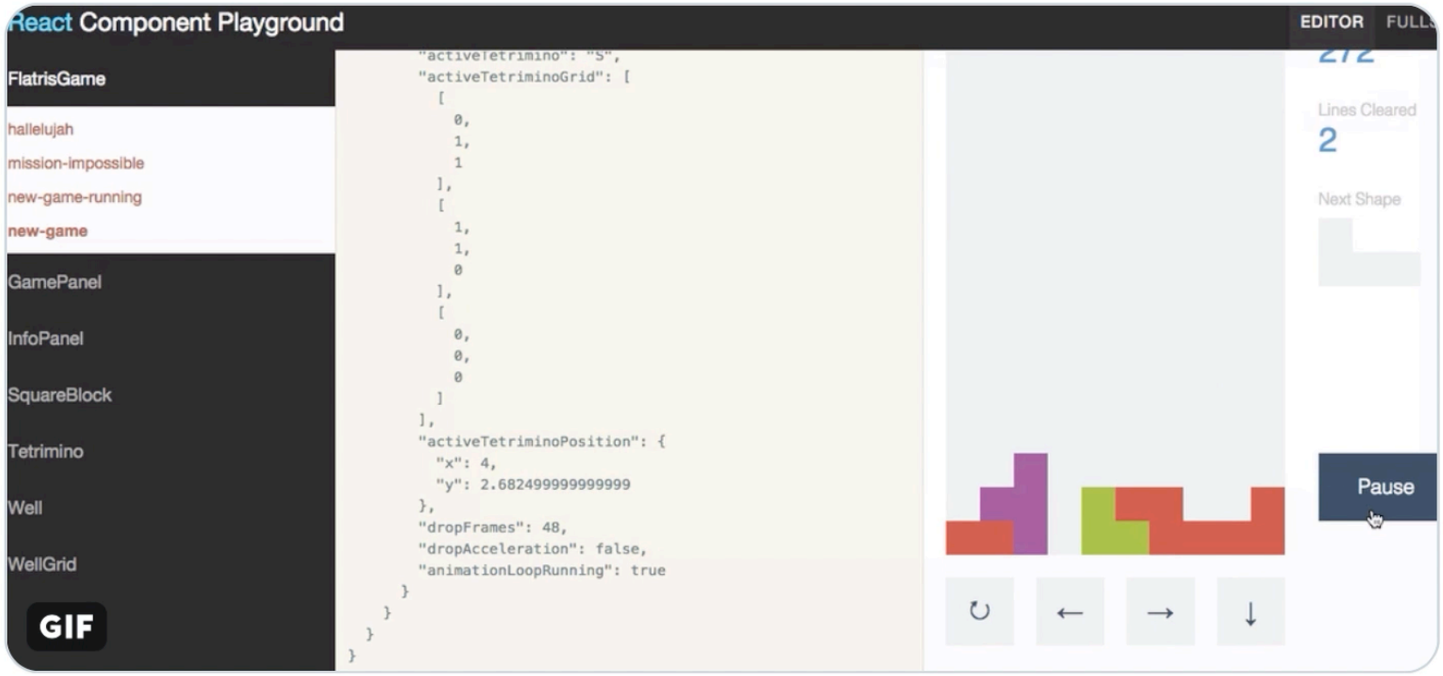
Pause


```
{
  "component": "Tetris",
  "state": {
    "gamePlaying": true,
    "gamePaused": false,
    "score": 20,
    "lines": 0,
    "nextTetrimino": "J"
  }
}
```

```
export default {
  component: SearchBox,
  state: {
    searchQuery: 'Who let the dogs out?'
  }
};
```

Ovidiu
@skidding

Behold the state of your @reactjs components, evolving in real time! —New in Cosmos 0.6.3
github.com/skidding/cosmos



6:31 PM · May 17, 2015 · Twitter Web Client

View Tweet activity

66 Retweets **141** Likes

Dan Abramov @dan_abramov · May 18, 2015
Replying to @skidding
@skidding @reactjs You're doing important work. Keep rocking.

2 3

```
export default {
  component: MyComponent,
  reduxState: {
    user: {
      //  ...
    }
  }
};
```

```
export default {
  component: MyComponent,
  url: '/users/5',
  route: '/users/:userId'
};
```

```
export default {
  component: MyComponent,
  fetch: [{
    matcher: '/users',
    response: [
      { id: 1, name: 'Prabu' },
      { id: 2, name: 'Karla' },
      { id: 3, name: 'Linbaba' }
    ]
  }]
};
```

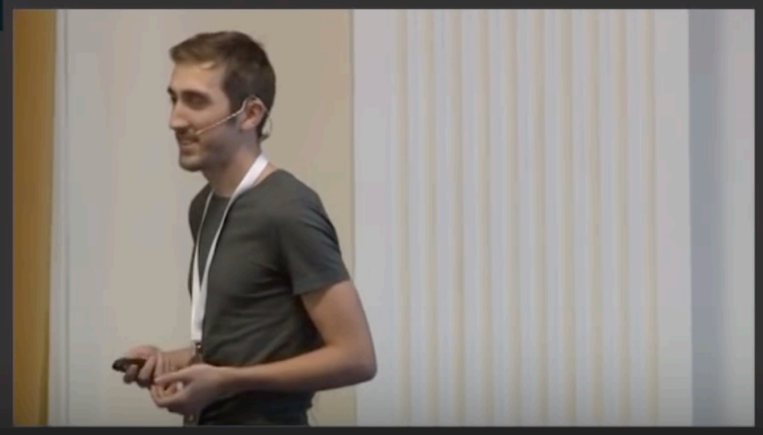
```
export default {
  component: MyComponent,
  localStorage: {
    userToken: 'foobar-token'
  }
};
```

JSHeroes

8 - 9 June 2017, Cluj-Napoca, Romania

UI development made simple

Ovidiu Chereches @skidding



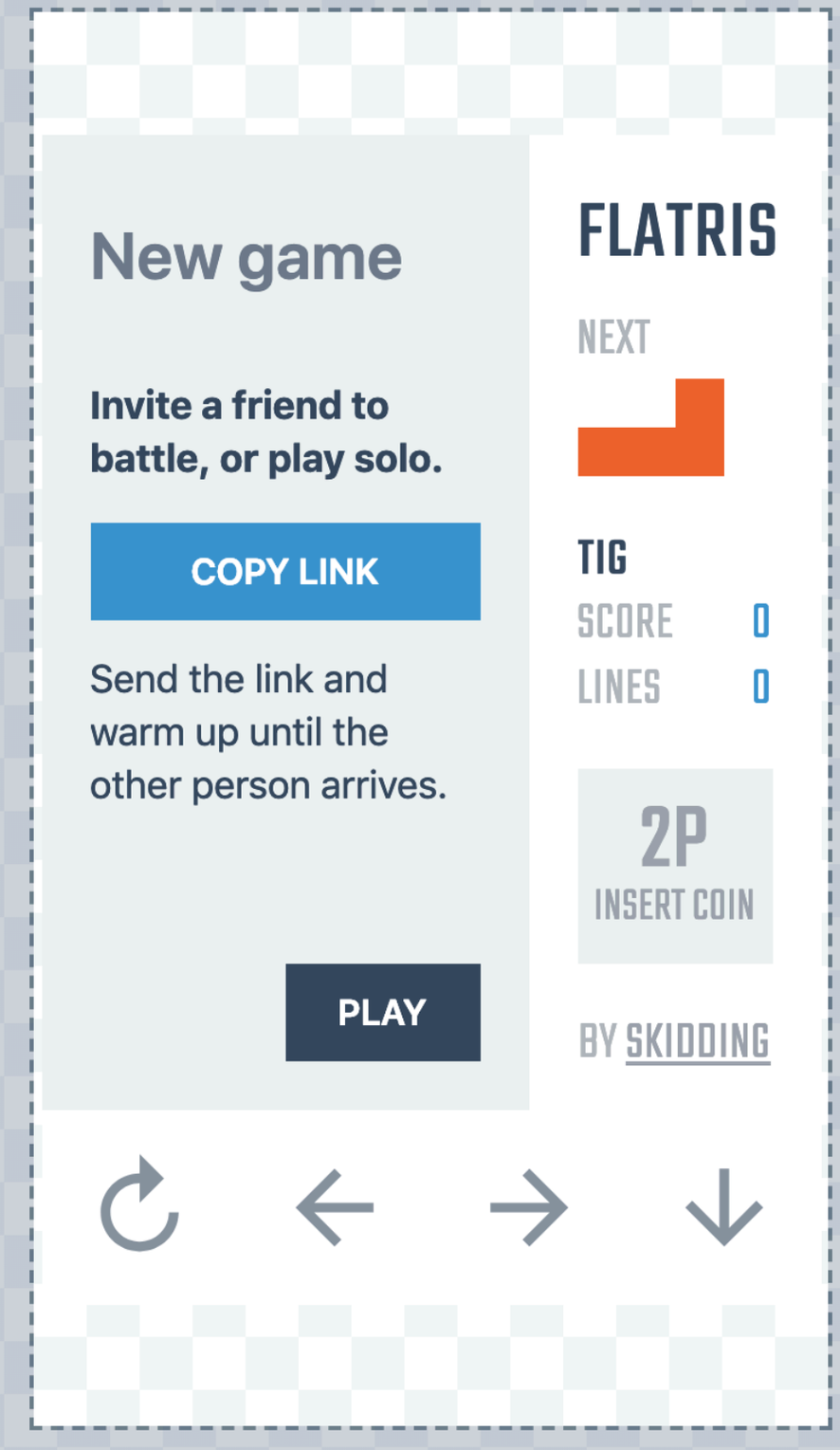
```
export default {
  component: Author,
  props: {
    authorId: 123
  },
  apollo: {
    resolveWith: {
      author: {
        __typename: 'Author',
        id: 123,
        firstName: 'Ovidiu'
      }
    }
  }
};
```

```
export default {
  component: MyComponent,
  viewport: {
    width: 375,
    height: 667
  }
};
```

Custom | iPhone 5 320 x 568 | iPhone 6 375 x 667 | iPhone 6 Plus 414 x 736 (Scaled 98%) | Medium 1024 x 768 (Scaled 88%) | Large 1440 x 900 (Scaled 62%) | 1080p 1920 x 1080 (Scaled 47%)

Search...

- controls
- effects
- pages
- screens
- Button
- CopyButton
- CopyGameLinkButton
- FlatrisGame Connect, SocketC...
  - double-hallelujah
  - enemy-hallelujah
  - hallelujah
  - hallelujah-guest
  - mission-impossible
  - new-game
  - new-game-running
- GameContainer
- GamePanel
- GamePreview
- GamePreviewShell
- Loading
- Logo
- PlayerInfo
- PointerButton



```
import createTestContext from 'react-cosmos-
import fixture from './fixture';

const { mount, getWrapper } = createTestContext({ fixture });

beforeEach(mount);

test('renders hello', () => {
  expect(getWrapper().text()).toContain('Hello World');
});
```

PRESENT

```
export default {
  component: Button,
  props: {
    children: 'Press me will you',
    disabled: true,
    onClick: () ⇒ console.log('Clicked!')
  }
}
```

```
export default (
  <Button disabled onClick={() ⇒ console.log('Clicked!')}>
    Press me will you
  </Button>
)
```

```
export default (
  <Wrapper>
    <Button disabled onClick={() ⇒ console.log('Clicked!')}>
      Press me will you
    </Button>
  </Wrapper>
)
```

```
export default (
  <Wrapper>
    <MyBenz />
    <MyOtherBenz />
    <MyOtherOtherBenz />
  </Wrapper>
)
```

```
export default {
  'primary enabled': <Button>Click me</Button>,
  'primary disabled': <Button disabled>Click me</Button>,
  'secondary enabled': <Button secondary>Click me</Button>,
  'secondary disabled': <Button secondary disabled>Click me</Button>
}
```

skidding Add note on delayed ref calls with StateMock d8f88c9 on Nov 15, 2018

1 contributor

217 lines (151 sloc) | 6.5 KB

Raw | Blame | History

React mock

build passing | codecov 93%

Declarative mocks for React state and global APIs.

Jump to

- **Why?**
- Component state
- Fetch requests
- LocalStorage
- XHR requests
- **How to contribute**

Why?

```
import { StateMock } from '@react-mock/state';

export default (
  <StateMock state={{ count: 5 }}>
    <Counter />
  </StateMock>
);
```

```
import { FetchMock } from '@react-mock/fetch';

export default (
  <FetchMock options={{ matcher: '/login', response: 401, method: 'POST' }}>
    <MyComponent />
  </FetchMock>
);
```

```
import { LocalStorageMock } from '@react-mock/localstorage';

export default (
  <LocalStorageMock items={{ sessionId: 're4lt0k3n' }}>
    <MyComponent />
  </LocalStorageMock>
);
```

```
import { MemoryRouter, Route } from 'react-router';

export default (
  <MemoryRouter initialEntries={[`/users/5`]}>
    <Route path="/users/:userId">
      <UserWithRouter />
    </Route>
  </MemoryRouter>
);
```

```
import { createStore } from 'redux';
import { Provider } from 'react-redux';

export default (
  <Provider store={createStore(myReducer, initialState)}>
    <ConnectedCounter />
  </Provider>
);
```

Text with prop value is rendered

The component renders variable text based on a string prop. We test that the component renders the value of the passed prop.

test.js

```
it('renders personalized greeting', async () => {
  // Render new instance in every test to prevent leaking state
  const { getByText } = render(<HelloMessage name="Satoshi" />);

  await waitForElement(() => getByText(/hello Satoshi/i));
});
```

Callback fires on button click

The component receives a callback prop and renders a button. We test that the callback prop is called when the button is clicked.

test.js

```
it('calls "onClick" prop on button click', () => {
  // Render new instance in every test to prevent leaking state
  const onClick = jest.fn();
  const { getByText } = render(<Button onClick={onClick} />);

  fireEvent.click(getByText(/click me nao/i));
```

FUTURE

CONTROL PANEL

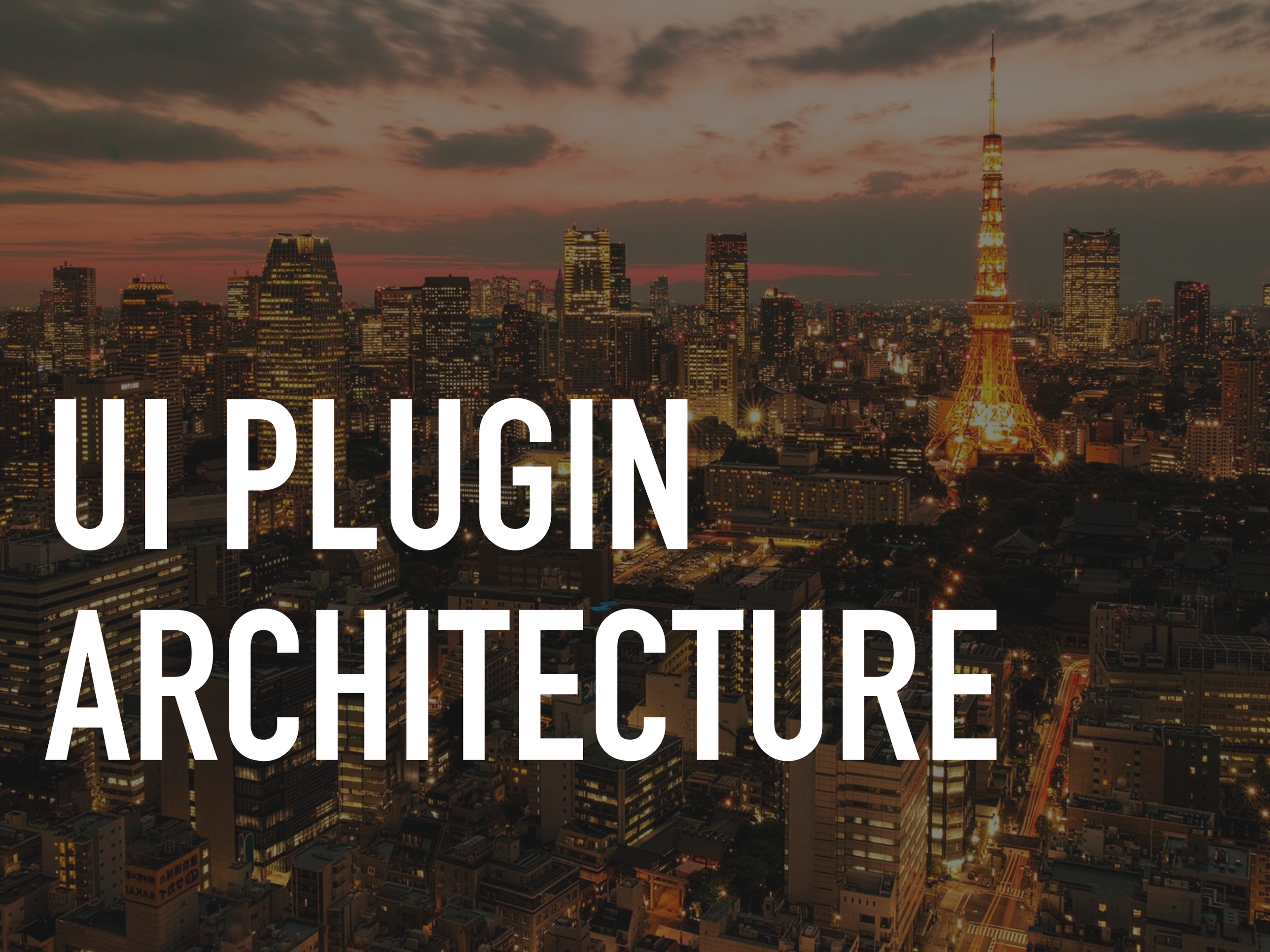
- "Fixture editor" for JSX fixtures
- Props reset vs transition
- Maybe: Capture hooks state

REDESIGN COSMOS DOCS

- Maybe: Generate docs website from decentralized README files
- Maybe: Embed docs in Cosmos UI

FIXTURE “VARIATIONS” / “SNAPSHOTS”

- Premise: Create first component fixture by hand, generate the rest through the Cosmos UI



UI PLUGIN ARCHITECTURE

ui-plugin

API for composable 3rd party UI plugins

Goal

Enable a system of user interface plugins that meets the following criteria:

- Plugins are authered, compiled and deployed independently, by anyone under any repository.
- Plugins can also be authored locally, sharing modules and compilation with the host project.
- Plugins can be installed at run time. This includes download from an external repository like npm.
- Plugins can be enabled and disabled at run time, including newly installed ones. No restart of the host app required.
- Plugins can manage visual components, app state, or both.
- Plugins can emit events that other plugins can subscribe to.
- Plugins expose public methods that other plugins can call synchronously.
- Each plugin has read-write access to own its state.
- Plugin state is private. State can be made available to other plugins though methods.
- Plugins can define render slots that other plugins can plug into. More plugins can take up the same slot and can choose to compose or override each other.
- Render slots derive their inputs from plugin state and method APIs, and update automatically whenever the state of a relevant plugin changes.
- Renders slots are free to manage their own private state that isn't accessible at plugin level.
- Plugin API is type-friendly.
- Plugins can share common libraries (eg. React) by configuring the host app to attach those libraries to the global namespace and configuring the plugin build to map related imports to the global namespace.

react-cosmos / react-cosmos

Unwatch 63 | Star 4,782 | Fork 212

Code | Issues 37 | Pull requests 1 | Projects 0 | Wiki | Insights | Settings

Branch: master

Create new file | Upload files | Find file | History

react-cosmos / packages / react-cosmos-playground2 / src / plugins /

skidding feat: Enable multiple (synced) web renderer connections (#933) Latest commit 561f792 4 days ago

..

ContentOverlay	feat: Add support for multiple (named) fixtures per file (#931)	7 days ago
ControlPanel	feat: Enable multiple (synced) web renderer connections (#933)	4 days ago
Core	feat: Enable multiple (synced) web renderer connections (#933)	4 days ago
Nav	Fix 404 state when fixture name doesn't match fixture path (#932)	6 days ago
Notifications	feat: Enable multiple (synced) web renderer connections (#933)	4 days ago
RendererCore	feat: Enable multiple (synced) web renderer connections (#933)	4 days ago
RendererHeader	feat: Enable multiple (synced) web renderer connections (#933)	4 days ago
RendererPreview	feat: Enable multiple (synced) web renderer connections (#933)	4 days ago
RendererRemote	feat: Enable multiple (synced) web renderer connections (#933)	4 days ago
ResponsivePreview	feat: Enable multiple (synced) web renderer connections (#933)	4 days ago
Router	feat: Add support for multiple (named) fixtures per file (#931)	7 days ago
Storage	Convert Playground2 to TypeScript (#929)	19 days ago

skidding feat: Enable multiple (synced) web renderer connections (#933) 561f792 4 days ago

1 contributor

33 lines (31 sloc) | 998 Bytes

Raw Blame History

```
import { StateUpdater } from 'react-cosmos-shared2/util';
import { FixtureState } from 'react-cosmos-shared2/fixtureState';
import {
  RendererId,
  FixtureNamesByPath,
  RendererRequest,
  RendererResponse
} from 'react-cosmos-shared2/renderer';

export type RendererCoreSpec = {
  name: 'rendererCore';
  state: {
    connectedRendererIds: RendererId[];
    primaryRendererId: null | RendererId;
    fixtures: FixtureNamesByPath;
    fixtureState: null | FixtureState;
  };
  methods: {
    getConnectedRendererIds(): RendererId[];
    getPrimaryRendererId(): null | RendererId;
    getFixtures(): FixtureNamesByPath;
    getFixtureState(): null | FixtureState;
    isRendererConnected(): boolean;
    isValidFixtureSelected(): boolean;
    setFixtureState(stateChange: StateUpdater<null | FixtureState>): void;
    selectPrimaryRenderer(primaryRendererId: RendererId): void;
    receiveResponse(msg: RendererResponse): void;
  };
  events: {
    request: (msg: RendererRequest) => void;
  };
};
```

```
render(<Slot name="root" />, document.getElementById('root'))
```

HOW CAN YOU HELP?

- Use Cosmos Next and give feedback
- Create upgrade code mods

```
export default {
  component: Button,
  props: {
    children: 'Press me will you',
    disabled: true,
    onClick: () ⇒ console.log('Clicked!')
  }
}
```

```
export default (
  <Button disabled onClick={() ⇒ console.log('Clicked!')}>
    Press me will you
  </Button>
)
```

NEW GAME

GITHUB

33 048 games 4 987 679 seconds played

2 945 684 lefts 3 338 286 rights 2 417 823 drops 2 631 244 rotations

560 446 lines cleared

Create a game and break the silence!

FLATRIS

NEXT

1P
INSERT COIN

2P
INSERT COIN

Tweets	Following	Followers	Likes
2,258	367	1,257	3,227

Ovidiu

@skidding Follows you

Building user interfaces and making UI development fun with @ReactCosmos
github.com/skidding

Cluj-Napoca, România

ovidiu.ch

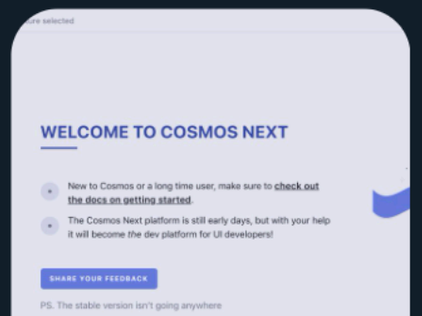
Joined September 2008

Tweets | Tweets & replies | Media

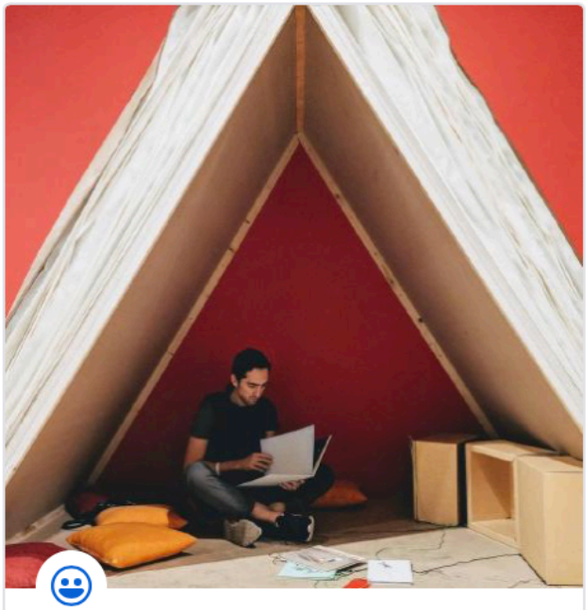
Pinned Tweet

Ovidiu @skidding · Jan 22

I worked my ass off to ship this redesign so p
about it! 🙏



React Cosmos @React
Cosmos 4.7 is out and y
🎉



Set your status

Ovidiu Cherecheș
skidding

PRO

Building user interfaces and making UI development fun.

Bucharest
skidding@gmail.com
https://twitter.com/skidding

Edit

Organizations

Overview | Repositories 98 | Projects 0 | Stars 204 | Followers 440 | Following 1

Pinned repositories

Customize your pinned repositories

react-cosmos/react-cosmos
Dev tool for creating reusable React components
TypeScript ★ 4.8k 212

illustrated-algorithms
Interactive algorithm visualizations
JavaScript ★ 2.5k 147

flatris
Fast-paced two-player web game
JavaScript ★ 874 153

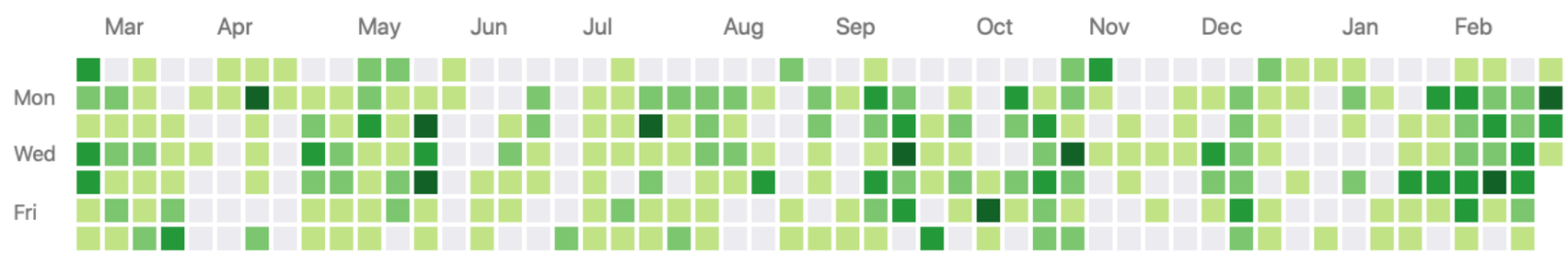
jobs-done
Ritual app for ending the work day inspired by Deep Work
JavaScript ★ 926 47

dragdealer
Drag-based vanilla JS component
JavaScript ★ 1k 149

hootsuite/grid
Drag and drop library for two-dimensional, resizable and responsive lists
JavaScript ★ 3.5k 297

1,433 contributions in the last year

Contribution settings



Learn how we count contributions.

Less More

BEFORE YOU GO...