Serious Functional Programming in TypeScript:

Putting Category Theory to Work for You

Christian Di Lorenzo

Data Science Engineer • JupiterOne

https://l.rcd.zone/news-repo

LIFEOMIC

FAST COMPANY

2019 Fast Company - Best Workplaces for Innovators

Outside

2019 Outside Magazine – The 50 Best Places to Work

Objectives

- Evaluate software correctness and reliability
- Observe “computational patterns”
- Synthesize mathematics and computer science

Top News in US

Britain sets December date for an election in which Brexit will dominate - The Washington Post

Karla Adam, William Booth • washingtonpost.com • an hour ago

Voters will be offered some stark choices over Brexit alongside the usual enthusiastic promises, scary scenarios, misrepresentations and foggy numbers. The Conservative Party under Johnson will run as the 100 percent for Brexit party, under the banner, Lets ...

NFL Trade Deadline Tracker 2019: Rams move on from Aqib Talib, but other rumored deals don't materialize - CBSSports.com

Tyler Sullivan • cbssports.com • an hour ago

The NFL's midseason game of musical chairs wrapped up on Tuesday afternoon at 4 p.m. ET with the trade deadline. Leading up to this year's deadline, we'd already seen a solid amount of movement throughout the league. Some of the headliners included Los Angel...

How to share your audio in iOS 13 - The Verge

Jay Peters • theverge.com • an hour ago

It works with AirPods and Beats headphones Last year, when I

News API

Get started | Documentation | News sources | Pricing | rcddeveloper@icloud.com

Search worldwide news with code

Get breaking news headlines, and search for articles from over 30,000 news sources and blogs with our news API

Get API key

All articles about Bitcoin from the last month, sorted by recent first

GET

https://newsapi.org/v2/everything?q=bitcoin&from=2019-09-29&sortBy=publishedAt&apiKey=a68e3d42763c4bfb958805aeee38b9a4


```
{
  status: "ok",
  totalResults: 4427,
```

Roadmap

1. Introduction to problem space
2. Approaches to computational abstractions
3. Program correctness and reliability
4. Functors, monoids, and monads
5. Putting it all together

1. Introduction to Problem Space

Mathematics vs. Programming? It's complicated.

$$f(x) = x^2 + 4x$$

```
const sum = (x: number) => {
  let result = 0;

  for (let i = 1; i <= x; i++) {
    result += i;
  }

  return result;
}
```

Composition: The Building Blocks

$$f \circ g = h$$

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "h"'] & B \arrow[dl, "g"] \\ C & \end{tikzcd}$$

Composition: The Building Blocks

```
1 const addOne = (x: number) => x + 1;
2 const subtractOne = (x: number) => x - 1;
3
4 const compose = <A, B, C>(g: (input: A) => B, f: (input: B) => C) => {
5   return (input: A) => f(g(input));
6 }
7
8 const addTwo = compose(addOne, addOne);
9 const subtractTwo = compose(subtractOne, subtractOne);
10 const identity = compose(addOne, subtractOne);
11
12 console.log(addTwo(0));      // => 2
13 console.log(identity(0));    // => 0
14 console.log(subtractTwo(0)); // => -2
```

2. Approaches to Computational Abstractions

Procedural

```
1 // ...
2 } else {
3   for (const e of entities) {
4     let entityId;
5     let entityIds;
6
7     if (e.entityId) {
8       entityId = e.entityId;
9     } else if (e.entityKey) {
10      const query = `Find * with _key='${e.entityKey}'`;
11      const res = await client.queryV1(query);
12      if (res.length === 1) {
13        entityId = res[0].entity._id;
14      } else if (res.length === 0) {
15        console.log(`Skipping entity with _key='${e.entityKey}' - NOT FOUND`);
16        continue;
17      } else if (res.length > 0) {
18        if (operation !== "delete" && !program.deleteDuplicates) {
19          console.log(`Skipping '${e.entityKey}' - KEY NOT UNIQUE`);
20          continue;
21        }
22        entityIds = res.map(r => r.entity._id);
23      } else {
24        console.log(`Skipping entity with _key='${e.entityKey}'`);
25        continue;
26      }
27      // ..
```

Object-Oriented Programming

```
1  export class ArticleList extends AbstractList<Article> {
2    constructor(props: any) {
3      super(props);
4      this.loadData = apiCall;
5    }
6
7    renderItem(article: Article, index: number) {
8      return (
9        <article>
10         <a href={article.url}>{article.title}</a>
11       </article>
12     );
13   }
14 }
```

Object-Oriented Programming

```
1 export class AbstractList<T> extends React.Component<{}, AbstractListState<T>>  {
2   loadData: () => Promise<T[]>;
3
4   constructor(props: any) {
5     super(props);
6     this.state = { items: [] };
7     this.loadData = () => Promise.resolve([]);
8   }
9
10  componentDidMount() {
11    this.loadData().then(items => this.setState({ items }));
12  }
13
14  renderItem(item: T, index: number) {
15    return (<span>TODO: Implement renderItem</span>);
16  }
17
18  render() {
19    return (
20      <div>{this.state.items.map((item, index) => this.renderItem(item, index))}</div>
21    );
22  }
23 }
```

Functional Programming

```
1 const loadInitialEvidenceData = async (input: CollectEvidenceBaseInput) => {
2   const dataLoader = /* ... */;
3
4   return Promise.all([
5     loadQuestions(input, dataLoader),
6     loadSpec(input, dataLoader),
7     loadSecurityItems(input, dataLoader),
8     loadControlsMapping(input, dataLoader),
9     loadRequirements(input, dataLoader)
10      .then(requirements => ({ requirements }))
11      .then(loadNotes(input, dataLoader))
12      .then(loadUserEvidence(input, dataLoader))
13  ]).then(([questions, spec, securityItems, controlsMapping, remaining]) => ({
14    ...remaining,
15    questions,
16    spec,
17    securityItems,
18    controlsMapping
19  }));
20 };
```

3. Program Correctness and Reliability

Feature: Compact Article Subtitle


```
1 {
2   "source": {
3     "id": null,
4     "name": "Space.com"
5   },
6   "author": "Mike Wall",
7   "title": "Mars Rover Curiosity Snaps Beautiful Selfi
  Space.com",
8   "description": "Curiosity did some \"wet chemistry\"
9   "url": "https://www.space.com/mars-rover-curiosity-s
10  "urlToImage": "https://cdn.mos.cms.futurecdn.net/8qF
11  "publishedAt": "2019-10-29T22:06:00Z",
12  "content": "NASA's Curiosity rover took this selfie
  sol, of its mission. The rover drilled twice in this l
  Etive.\"\r\nNASA's Curiosity Mars rover\r\n performed some rare science work recently, the… [+3493
  chars]"
13 }
```


Mars Rover Curiosity Snaps Beautiful Selfie After Rare Chemistry Experiment - Space.com

Mike Wall • space.com • 3 hours ago

NASA's Curiosity rover took this selfie on Oct. 11, 2019, the 2,553rd Martian day, or sol, of its mission. The rover drilled twice in this location, which is nicknamed "Glen Etive." NASA's Curiosity Mars rover performed some rare science work recently, the...

Fetching Articles

```
1  import axios from 'axios';
2
3  const API_KEY = /* actual api key */;
4
5  const loadTopHeadlines = async (): Promise<Article[]> => {
6    const response = await axios.get(
7      `https://newsapi.org/v2/top-headlines?country=us&apiKey=${API_KEY}`
8    );
9
10   return response.data.articls as Article[];
11 };
```

Rendering Subtitles

```typescript
interface Article {
  title: string;
  url: string;
  author: string;
  content: string;
  publishedAt: string;
}

const subtitles = (articles: Article[]) => {
  return articles.map(article => {
    const domain = article.url.match(/https?:\/\/w?w?w?\.?([\w\.]+)[\/$]{1}/)![1];
    const relativeTime = dayjs(new Date(article.publishedAt)).fromNow();

    return [article.author, domain, relativeTime].join(' • ');
  });
};
```

Nick Kostov • wsj.com • 3 hours ago

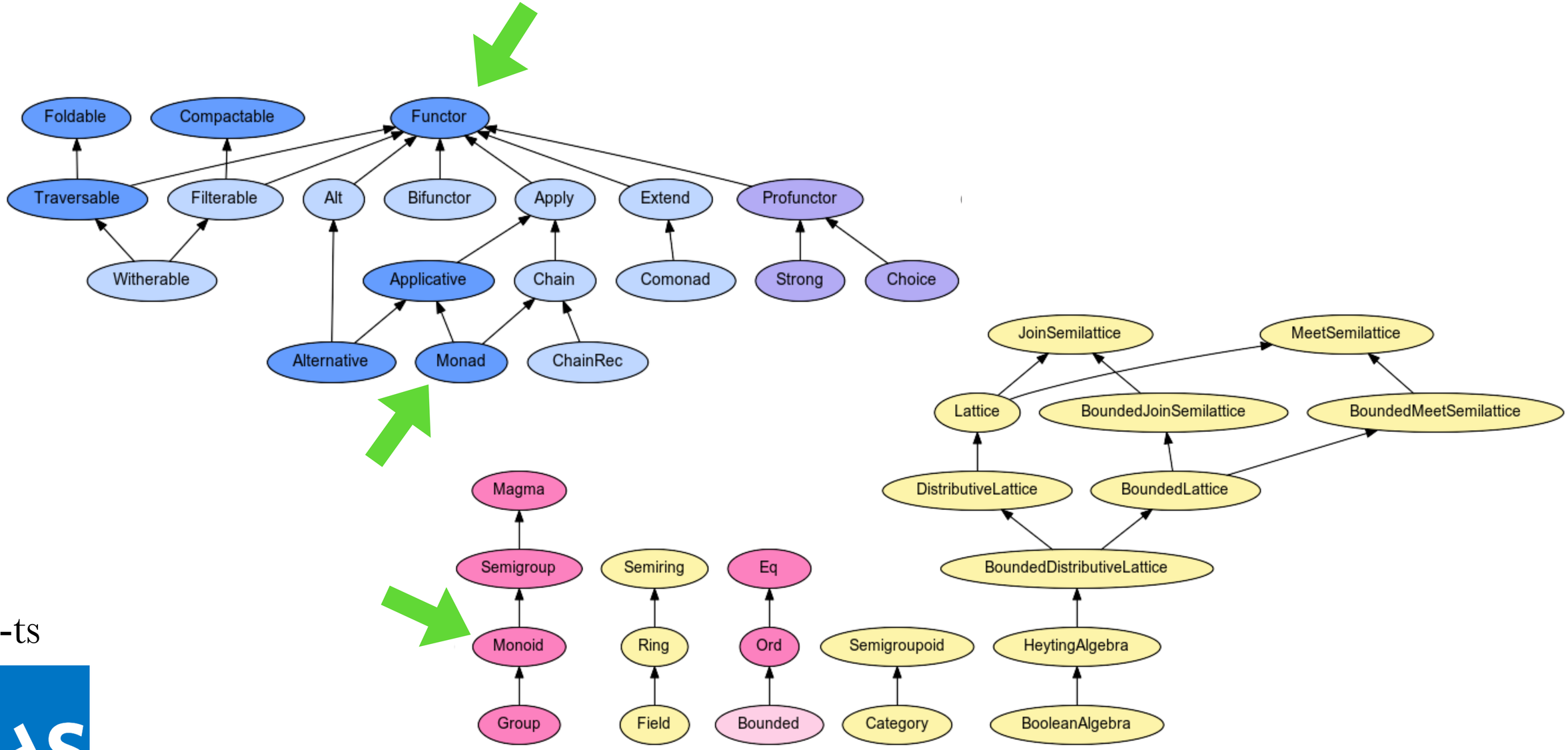
nytimes.com • 3 hours ago

Cheat Preview: Solution

```
1 interface Article {
2   title: string;
3   url: string;
4   author: Option<string>;
5   content: Option<string>;
6   publishedAt: Date;
7 }
```

```
1 export const articleSubtitle = (article: Article): Option<string> => {
2   const elements = [
3     article.author,
4     articleDomainName(article),
5     some(articlePublishedAt(article))
6   ];
7
8   return pipe(
9     elements
10    A.map(chain(fromEmptyNullable)),
11    A.compact,
12    join(' • '),
13    fromEmptyNullable
14  );
15 };
```

4. Functors, Monoids, and Monads



fp-ts

https://gcanti.github.io/fp-ts/

4a. Functors

Functors, Monoids, and Monads

Functors (Haskell)

```haskell
class Functor f where
    fmap :: (a -> b) -> f a -> f b

-- Example implementation: Maybe
data Maybe a = Just a | Nothing

instance Functor Maybe where
  fmap func (Just value) = Just (func value)
  fmap func Nothing = Nothing

-- Example usage
fmap (+1) [1, 2, 3] -- => [2, 3, 4]
fmap (*2) (Just 3)  -- => Just 6
fmap (*4) Nothing   -- => Nothing

-- Alternate Syntax
(+1) <$> [1, 2, 3]
(*2) <$> (Just 3)
```

Functors (TypeScript)

```typescript
// Functor.ts
// Instances must satisfy the following laws:
//    Identity:
//      F.map(fa, a => a) = fa
//    Composition:
//      F.map(fa, a => bc(ab(a))) = F.map(F.map(fa, ab), bc)
export interface Functor<F> {
  readonly URI: F
  readonly map: <A, B>(fa: HKT<F, A>, f: (a: A) => B) => HKT<F, B>
}

// Array.ts (converted to be pipeable)
export const map = <A, B>(f: (a: A) => B) => (fa: A[]) => B[];
```

Usage

```
1  export const articleSubtitle = (article: Article): Option<string> => {
2    const elements = [
3      article.author,
4      articleDomainName(article),
5      some(articlePublishedAt(article))
6    ];
7
8    return pipe(
9      elements
10     A.map(chain(fromEmptyNullable)),
11     A.compact,
12     join(' • '),
13     fromEmptyNullable
14   );
15 };
```

Nick Kostov • wsj.com • 3 hours ago

nytimes.com • 3 hours ago

4b. Monoids

Functors, Monoids, and Monads

Monoids (Haskell)

```haskell
class Monoid m where
  mempty :: m
  mappend :: m -> m -> m

  mconcat :: [m] -> m
  mconcat = foldr mappend mempty

-- Example implementation: Array
instance Monoid [a] where
  mempty = []
  mappend = (++)

-- Example implementation: Sum
newtype Sum n = Sum n

instance Monoid (Sum n) where
  mempty = 0
  mappend = (+)

-- Example implementation: String
instance Monoid String where
  mempty = ""
  mappend = (++)

-- Example usage
mconcat [(Sum 1), (Sum 2)] -- => (Sum 3)
"Hello" <> " world!"       -- => "Hello world!"
```

Monoids (TypeScript)

```typescript
// Magma.ts, Semigroup.ts, Monoid.ts
export interface Magma<A> {
  readonly concat: (x: A, y: A) => A
}

export interface Semigroup<A> extends Magma<A> {}

export interface Monoid<A> extends Semigroup<A> {
  readonly empty: A
}
```

```typescript
// Url.ts
const join = (sep: string) => (items: string[]) =>
  intercalate(monoidString, array)(sep, items);

const Url: Monoid<string> = {
  empty: '',
  concat: (left, right) => pipe(
    [left, right],
    map(trimSlash),
    compact,
    join('/')
  )
};

Url.concat('https://newsapi.org', 'v2')
// => https://newsapi.org/v2

Url.concat('https://newsapi.org/', '')
// => https://newsapi.org
```

4c. Monads

Functors, Monoids, and Monads

Monads (Haskell)

```haskell
class Monad m where
  (>>=)  :: m a -> (  a -> m b) -> m b
  -- ...

-- Example implementation: Maybe
instance Monad (Maybe x) where
  (Just x) >>= k  = k x
  Nothing  >>= _  = Nothing
  -- ...

lastName :: String -> Maybe String
-- ...implementation of lastName

-- Example usage
(Just "Christian Di Lorenzo") >>= lastName
-- => (Just "Di Lorenzo")

Nothing >>= lastName
(Just "Joy") >>= lastName
-- => Nothing
```

Monads (TypeScript)

```typescript
// Apply.ts, Chain.ts, Applicative.ts, Monad.ts
export interface Apply<F> extends Functor<F> {
  readonly ap: <A, B>(fab: HKT<F, (a: A) => B>, fa: HKT<F, A>) => HKT<F, B>
}

export interface Chain<F> extends Apply<F> {
  readonly chain: <A, B>(fa: HKT<F, A>, f: (a: A) => HKT<F, B>) => HKT<F, B>
}

export interface Applicative<F> extends Apply<F> {
  readonly of: <A>(a: A) => HKT<F, A>
}

export interface Monad<F> extends Applicative<F>, Chain<F> {}
```

Usage

```
1  export const articleSubtitle = (article: Article): Option<string> => {
2    const elements = [
3      article.author,
4      articleDomainName(article),
5      some(articlePublishedAt(article))
6    ];
7
8    return pipe(
9      elements
10     A.map(chain(fromEmptyNullable)),
11     A.compact,
12     join(' • '),
13     fromEmptyNullable
14   );
15 };
```

Nick Kostov • wsj.com • 3 hours ago

nytimes.com • 3 hours ago

5. Putting It All Together

Demo

Links

- “fp-ts”: https://gcanti.github.io/fp-ts
- “io-ts-types”: https://gcanti.github.io/io-ts-types/docs/modules
- Functors, Applicatives, and Monads: http://adit.io/posts/2013-04-17-functors,_applicatives,_and_monads_in_pictures.html
- Monads (fastest): https://dev.to/bobbypriambodo/comment/j27
- Haskell: https://wiki.haskell.org
- *Category Theory* by Steve Awodey