



Video 3.6


React and APIs

Chris Murphy

Review

- React allows us to create modular JavaScript components that we can use in our HTML pages to develop web applications
- How can we take advantage of the modular nature of the Web?

Web development in the old days



```
19
20 main(int argc, char *argv[]) {
21     entry entries[10000];
22     register int x,m=0;
23     char *cl;
24
25     printf("Content-type: text/html%c%c",10,10);
26
27     if(strcmp(getenv("REQUEST_METHOD"),"GET")) {
28         printf("This script should be referenced with a METHOD of GET.\n");
29         printf("If you don't understand this, see this ");
30         printf("<A
31     HREF=\"http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/Docs/fill-out-forms/overview.html\">forms
32     overview</A>.%c",10);
33     exit(1);
34 }
35
36 cl = getenv("QUERY_STRING");
37 if(cl == NULL) {
38     printf("No query information to decode.\n");
39     exit(1);
40 }
41 for(x=0;cl[x] != '\0';x++) {
42     m=x;
43     getword(entries[x].val,cl,&');
44     plustospace(entries[x].val);
45     unescape_url(entries[x].val);
46     getword(entries[x].name,entries[x].val, '=');
47 }
48
49 printf("<H1>Query Results</H1>");
50 printf("You submitted the following name/value pairs:<p>%c",10);
51 printf("<ul>%c",10);
52
53 for(x=0; x <= m; x++)
54     printf("<li> <code>%s = %s</code>%c",entries[x].name,
55         entries[x].val,10);
56 printf("</ul>%c",10);
57 }
```

Designing software for the Web

- How can we make this better?
- Use design paradigms with the following goals:
 - Break up code into distinct components
 - The components interact with each other in a “standard” manner
 - Makes it easy to change any component
 - Less dependency and coupling

Service-Oriented Architecture

- Letter by Jeff Bezos to all employees at Amazon (circa 2002)
- “All teams will henceforth expose their data and functionality through service interfaces.”
- “The only communication allowed is via service interface calls over the network.”
- “All service interfaces, without exception, must be designed from the ground up to be externalizable. That is to say, the team must plan and design to be able to expose the interface to developers in the outside world. No exceptions.”

Software as a Service (SaaS)

- Extends the idea of Service-oriented Architecture
- SaaS – entire software runs as a service
- Examples
 - Google Docs (as opposed to Microsoft Word)
 - Gmail (as opposed to email clients like Outlook)
 - Cloud9 (as opposed to Eclipse)

Software as a Service: Advantages

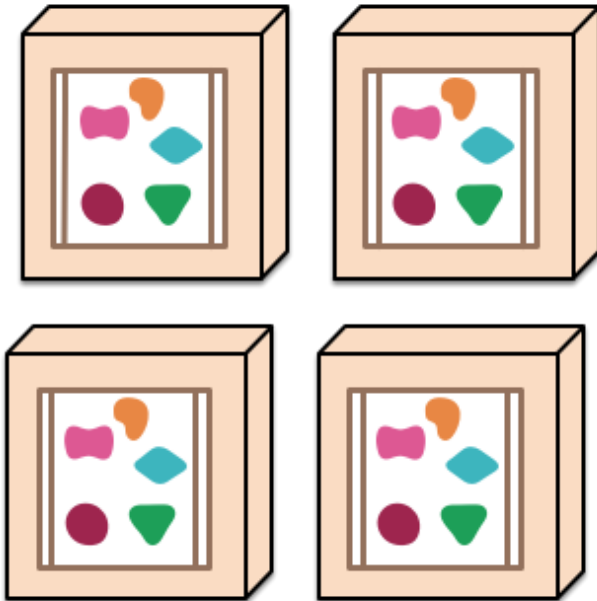
- No user installation is needed
- Less likely to lose data
- Users can collaborate with each other
- Upgrading software and datasets is easy
- Software is centralized in a single environment (don't need to worry about versions of operating systems, etc.)
- User's computer (hardware and software specifications) don't matter – usually all that's needed is a web browser

Microservices

A monolithic application puts all its functionality into a single process...



... and scales by replicating the monolith on multiple servers

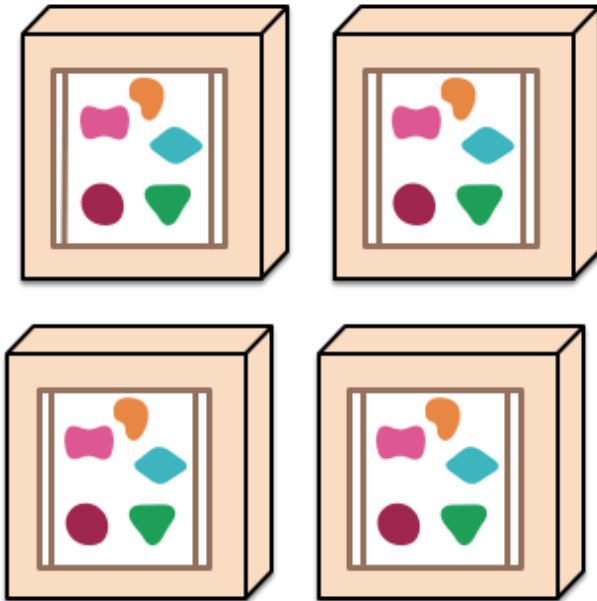


Microservices

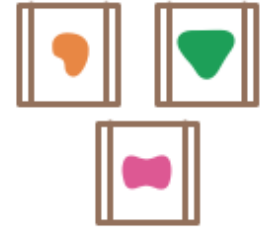
A monolithic application puts all its functionality into a single process...



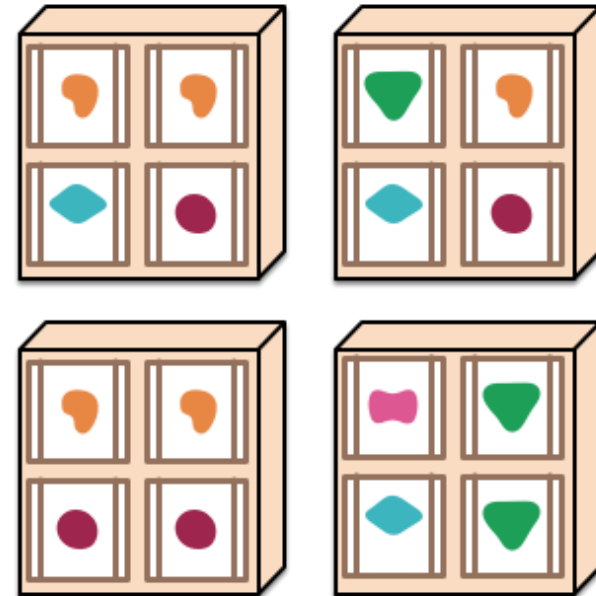
... and scales by replicating the monolith on multiple servers



A microservices architecture puts each element of functionality into a separate service...



... and scales by distributing these services across servers, replicating as needed.



REST

- How do well-designed web servers behave?
- Define a set of rules and conventions
- **REST:** REpresentational State Transfer
- Collection of *resources* on which specific operations can be performed
- URI names *resources*, not pages or actions
- Self-contained - which resource, what to do with it, server doesn't need to maintain state between requests

What is an API?

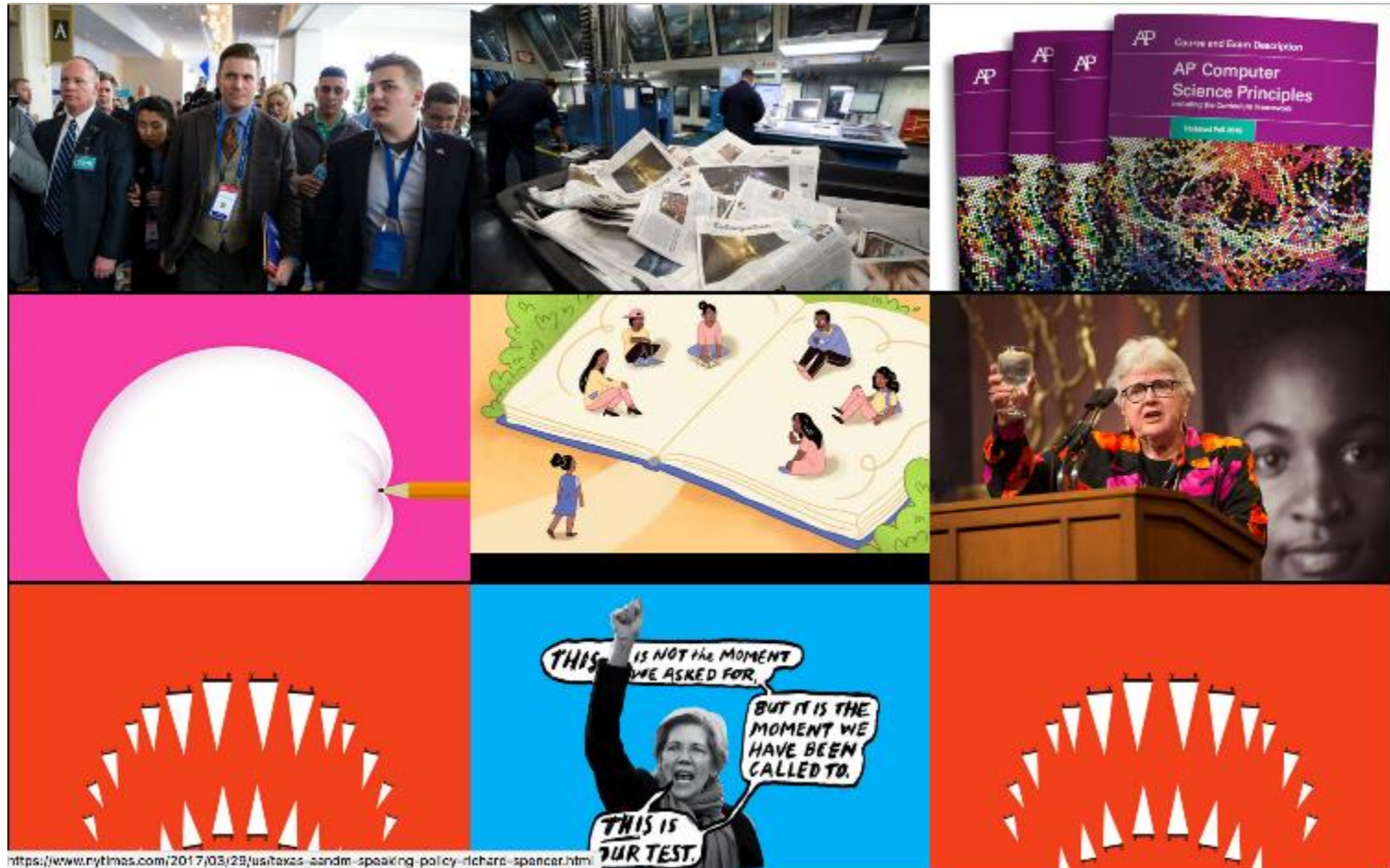
- An API is an **Application Programming Interface**
- In web programming, an API is a URL or a set of URLs that returns pure data to requests
- APIs can be used to incorporate data and functionality from other sources in your webapp

The New York Times

- *The New York Times* is an American daily newspaper company
- *The New York Times* provides a variety of APIs that provide article data, books data, movies data, and more
- You can see more about available APIs and request access at <https://developer.nytimes.com/>

Our Goal

- Created a webapp that creates a dashboard of clickable *New York Times* article images



Getting Started

- Request a developer key for the Article Search API at <https://developer.nytimes.com/>
- You can test out the API and what the returned data looks like at https://developer.nytimes.com/article_search_v2.json

Structure

- First, let's create a new React component named ArticlesGrid to store the data in the page

```
class ArticlesGrid extends React.Component {  
  constructor (props) {  
    super(props);  
    this.state = {  
      articles: []  
    };  
  }  
  
  . . .  
}
```

Structure

- First, let's create a new React component named ArticlesGrid to store the data in the page

```
class ArticlesGrid extends React.Component {  
  constructor (props) {  
    super(props);  
    this.state = {  
      articles: []  
    };  
  }  
  
  . . .
```


Structure

- First, let's create a new React component named ArticlesGrid to store the data in the page

```
class ArticlesGrid extends React.Component {  
  constructor (props) {  
    super(props);  
    this.state = {  
      articles: []  
    };  
  }  
  . . .  
}
```

Structure

- First, let's create a new React component named ArticlesGrid to store the data in the page

```
class ArticlesGrid extends React.Component {  
  constructor (props) {  
    super(props);  
    this.state = {  
      articles: []  
    };  
  }  
  
  . . .
```

Getting Data

- Let's start by getting some data from the API with a search term.
- We will do this with jQuery AJAX

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  componentDidMount () {  
    var url=  
      'http://api.nytimes.com/svc/search/v2/articlesearch.json?'  
      + 'api-key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';  
  
    $.getJSON(url, function(data, status) {  
      return this.setState({articles: this.parse(data)});  
    }).bind(this);  
  }  
  
  . . .  
}
```

Getting Data

- Let's start by getting some data from the API with a search term.
- We will do this with jQuery AJAX

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  componentDidMount () {  
    var url=  
      'http://api.nytimes.com/svc/search/v2/articlesearch.json?'  
      + 'api-key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';  
  
    $.getJSON(url, function(data, status) {  
      return this.setState({articles: this.parse(data)});  
    }).bind(this);  
  }  
  
  . . .  
}
```

Getting Data

- Let's start by getting some data from the API with a search term.
- We will do this with jQuery AJAX

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  componentDidMount () {  
    var url=  
      'http://api.nytimes.com/svc/search/v2/articlesearch.json?'  
      + 'api-key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';  
  
    $.getJSON(url, function(data, status) {  
      return this.setState({articles: this.parse(data)});  
    }).bind(this);  
  }  
  
  . . .  
}
```

Getting Data

- Let's start by getting some data from the API with a search term.
- We will do this with jQuery AJAX

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  componentDidMount () {  
    var url=  
      'http://api.nytimes.com/svc/search/v2/articlesearch.json?'  
      + 'api-key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';  
  
    $.getJSON(url, function(data, status) {  
      return this.setState({articles: this.parse(data)});  
    }).bind(this);  
  }  
  
  . . .  
}
```

Getting Data

- Let's start by getting some data from the API with a search term.
- We will do this with jQuery AJAX

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  componentDidMount () {  
    var url=  
      'http://api.nytimes.com/svc/search/v2/articlesearch.json?'  
      + 'api-key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';  
  
    $.getJSON(url, function(data, status) {  
      return this.setState({articles: this.parse(data)});  
    }).bind(this) ;  
  }  
  
  . . .  
}
```

Getting Data

- Let's start by getting some data from the API with a search term.
- We will do this with jQuery AJAX

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  componentDidMount () {  
    var url=  
      'http://api.nytimes.com/svc/search/v2/articlesearch.json?'  
      + 'api-key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';  
  
    $.getJSON(url, function(data, status) {  
      return this.setState({articles: this.parse(data)});  
    }).bind(this);  
  }  
  
  . . .  
}
```


Getting Data

- Let's start by getting some data from the API with a search term.
- We will do this with jQuery AJAX

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  componentDidMount () {  
    var url=  
      'http://api.nytimes.com/svc/search/v2/articlesearch.json?'  
      + 'api-key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';  
  
    $.getJSON(url, function(data, status) {  
      return this.setState({articles: this.parse(data)});  
    }).bind(this);  
  }  
  
  . . .  
}
```

Getting Data

- Let's start by getting some data from the API with a search term.
- We will do this with jQuery AJAX

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  componentDidMount () {  
    var url=  
      'http://api.nytimes.com/svc/search/v2/articlesearch.json?'  
      + 'api-key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';  
  
    $.getJSON(url, function(data, status) {  
      return this.setState({articles: this.parse(data)});  
    }).bind(this));  
  }  
  
  . . .  
}
```

Getting Data

- Let's start by getting some data from the API with a search term.
- We will do this with jQuery AJAX

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  componentDidMount () {  
    var url=  
      'http://api.nytimes.com/svc/search/v2/articlesearch.json?'  
      + 'api-key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';  
  
    $.getJSON(url, function(data, status) {  
      return this.setState({articles: this.parse(data)});  
    }).bind(this);  
  }  
  
  . . .  
}
```

Getting Data

- Let's start by getting some data from the API with a search term.
- We will do this with jQuery AJAX

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  componentDidMount () {  
    var url=  
      'http://api.nytimes.com/svc/search/v2/articlesearch.json?'  
      + 'api-key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';  
  
    $.getJSON(url, function(data, status) {  
      return this.setState({articles: this.parse(data)});  
    }).bind(this);  
  }  
  
  . . .  
}
```

Getting Data

- Let's start by getting some data from the API with a search term.
- We will do this with jQuery AJAX

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  componentDidMount () {  
    var url=  
      'http://api.nytimes.com/svc/search/v2/articlesearch.json?'  
      + 'api-key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';  
  
    $.getJSON(url, function(data, status) {  
      return this.setState({articles: this.parse(data)});  
    }).bind(this);  
  }  
  
  . . .  
}
```

Getting Data

- Let's start by getting some data from the API with a search term.
- We will do this with jQuery AJAX

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  componentDidMount () {  
    var url=  
      'http://api.nytimes.com/svc/search/v2/articlesearch.json?'  
      + 'api-key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';  
  
    $.getJSON(url, function(data, status) {  
      return this.setState({articles: this.parse(data)});  
    } .bind(this));  
  }  
  
  . . .  
}
```

```
{
  "response": {
    "meta": {
      "hits": 4251,
      "time": 36,
      "offset": 0
    },
    "docs": [
      {
        "web_url": "https://query.nytimes.com/gst/...",
        "snippet": "Having invited Senator LA FOLLETTE to make a
statement of his cause and case in today's TIMES, we would
refer to it only with becoming courtesy. Its great lack must
be obvious to every reader. This is its extreme indefiniteness.
The Senator uses w...",
        "lead_paragraph": "Having invited Senator LA FOLLETTE
to make a statement of his cause and case in today's TIMES,
we would refer to it only with becoming courtesy. Its great
lack must be obvious to every reader. This is its extreme
indefiniteness. The Senator uses with vehemence a great many
terms without defining one of them.",
        . . .
      }
    ]
  }
  {
    "web_url": "https://query.nytimes.com/gst/...",
    "snippet": "At Washington last week the spokesman..."
  }
}
```

```
{
  "response": {
    "meta": {
      "hits": 4251,
      "time": 36,
      "offset": 0
    },
    "docs": [
      {
        "web_url": "https://query.nytimes.com/gst/...",
        "snippet": "Having invited Senator LA FOLLETTE to make a
statement of his cause and case in today's TIMES, we would
refer to it only with becoming courtesy. Its great lack must
be obvious to every reader. This is its extreme indefiniteness.
The Senator uses w...",
        "lead_paragraph": "Having invited Senator LA FOLLETTE
to make a statement of his cause and case in today's TIMES,
we would refer to it only with becoming courtesy. Its great
lack must be obvious to every reader. This is its extreme
indefiniteness. The Senator uses with vehemence a great many
terms without defining one of them.",
        . . .
      }
    ]
  }
}
```



```
{
  "response": {
    "meta": {
      "hits": 4251,
      "time": 36,
      "offset": 0
    },
    "docs": [
      {
        "web_url": "https://query.nytimes.com/gst/...",
        "snippet": "Having invited Senator LA FOLLETTE to make a
statement of his cause and case in today's TIMES, we would
refer to it only with becoming courtesy. Its great lack must
be obvious to every reader. This is its extreme indefiniteness.
The Senator uses w...",
        "lead_paragraph": "Having invited Senator LA FOLLETTE
to make a statement of his cause and case in today's TIMES,
we would refer to it only with becoming courtesy. Its great
lack must be obvious to every reader. This is its extreme
indefiniteness. The Senator uses with vehemence a great many
terms without defining one of them.",
        . . .
      }
    ]
  }
  {
    "web_url": "https://query.nytimes.com/gst/...",
    "snippet": "At Washington last week the spokesman..."
  }
}
```

```
{
  "response": {
    "meta": {
      "hits": 4251,
      "time": 36,
      "offset": 0
    },
    "docs": [
      {
        "web_url": "https://query.nytimes.com/gst/...",
        "snippet": "Having invited Senator LA FOLLETTE to make a
statement of his cause and case in today's TIMES, we would
refer to it only with becoming courtesy. Its great lack must
be obvious to every reader. This is its extreme indefiniteness.
The Senator uses w...",
        "lead_paragraph": "Having invited Senator LA FOLLETTE
to make a statement of his cause and case in today's TIMES,
we would refer to it only with becoming courtesy. Its great
lack must be obvious to every reader. This is its extreme
indefiniteness. The Senator uses with vehemence a great many
terms without defining one of them.",
        . . .
      }
    ]
  }
  {
    "web_url": "https://query.nytimes.com/gst/...",
    "snippet": "At Washington last week the spokesman..."
  }
}
```

```
{
  "response": {
    "meta": {
      "hits": 4251,
      "time": 36,
      "offset": 0
    },
    "docs": [
      {
        "web_url": "https://query.nytimes.com/gst/...",
        "snippet": "Having invited Senator LA FOLLETTE to make a
statement of his cause and case in today's TIMES, we would
refer to it only with becoming courtesy. Its great lack must
be obvious to every reader. This is its extreme indefiniteness.
The Senator uses w...",
        "lead_paragraph": "Having invited Senator LA FOLLETTE
to make a statement of his cause and case in today's TIMES,
we would refer to it only with becoming courtesy. Its great
lack must be obvious to every reader. This is its extreme
indefiniteness. The Senator uses with vehemence a great many
terms without defining one of them.",
        . . .
      }
    ]
  },
  {
    "web_url": "https://query.nytimes.com/gst/...",
    "snippet": "At Washington last week the spokesman..."
  }
}
```

```
{
  "response": {
    "meta": {
      "hits": 4251,
      "time": 36,
      "offset": 0
    },
    "docs": [
      {
        "web_url": "https://query.nytimes.com/gst/...",
        "snippet": "Having invited Senator LA FOLLETTE to make a
statement of his cause and case in today's TIMES, we would
refer to it only with becoming courtesy. Its great lack must
be obvious to every reader. This is its extreme indefiniteness.
The Senator uses w...",
        "lead_paragraph": "Having invited Senator LA FOLLETTE
to make a statement of his cause and case in today's TIMES,
we would refer to it only with becoming courtesy. Its great
lack must be obvious to every reader. This is its extreme
indefiniteness. The Senator uses with vehemence a great many
terms without defining one of them.",
        ". . ."
      }
    ]
  }
}
```

```
{
  "response": {
    "meta": {
      "hits": 4251,
      "time": 36,
      "offset": 0
    },
    "docs": [
      {
        "web_url": "https://query.nytimes.com/gst/...",
        "snippet": "Having invited Senator LA FOLLETTE to make a
statement of his cause and case in today's TIMES, we would
refer to it only with becoming courtesy. Its great lack must
be obvious to every reader. This is its extreme indefiniteness.
The Senator uses w...",
        "lead_paragraph": "Having invited Senator LA FOLLETTE
to make a statement of his cause and case in today's TIMES,
we would refer to it only with becoming courtesy. Its great
lack must be obvious to every reader. This is its extreme
indefiniteness. The Senator uses with vehemence a great many
terms without defining one of them.",
        . . .
      }
    ]
  },
  {
    "web_url": "https://query.nytimes.com/gst/...",
    "snippet": "At Washington last week the spokesman..."
  }
}
```

Parsing Data

- The *NY Times* API will return a lot of raw data!
- Let's parse the data into something we can use
- For each article, we'll keep the image URL, title, and URL to original article if a large image exists

```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:    article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:   article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }

  isXL (image) {
    return image.subtype === 'xlarge';
  }
}

```

```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:    article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:   article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }
}

isXL (image) {
  return image.subtype === 'xlarge';
}

```



```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:    article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:   article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }

  isXL (image) {
    return image.subtype === 'xlarge';
  }
}

```

```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:   article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:  article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }

  isXL (image) {
    return image.subtype === 'xlarge';
  }
}

```

```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:   article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:  article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }

  isXL (image) {
    return image.subtype === 'xlarge';
  }
}

```

```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:    article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:   article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }

  isXL (image) {
    return image.subtype === 'xlarge';
  }
}

```

```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:   article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:  article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }

  isXL (image) {
    return image.subtype === 'xlarge';
  }
}

```

```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:   article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:  article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }

  isXL (image) {
    return image.subtype === 'xlarge';
  }
}

```

```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:    article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:   article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }
}

isXL (image) {
  return image.subtype === 'xlarge';
}

```

```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:   article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:  article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }

  isXL (image) {
    return image.subtype === 'xlarge';
  }
}

```



```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:    article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:   article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }

  isXL (image) {
    return image.subtype === 'xlarge';
  }
}

```

```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:    article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:   article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }

  isXL (image) {
    return image.subtype === 'xlarge';
  }
}

```

```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:    article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:   article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }

  isXL (image) {
    return image.subtype === 'xlarge';
  }
}

```

```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:    article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:    article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }

  isXL (image) {
    return image.subtype === 'xlarge';
  }
}

```

```

class ArticlesGrid extends React.Component {
  . . .

  parse(results) {
    if( !results || !results.response ) return [];
    var articles = results.response.docs;
    var parsedArticles = [];
    for (var i = 0; i < articles.length; i++) {
      var article = articles[i];
      if (article.multimedia.find(this.isXL)) {
        parsedArticles.push({
          id:      article._id,
          title:    article.headline.main || 'Untitled',
          imageURL: article.multimedia.find(this.isXL).url || '#',
          webURL:   article.web_url || '#'
        });
      }
    }
    return parsedArticles;
  }

  isXL (image) {
    return image.subtype === 'xlarge';
  }
}

```

Displaying Data

- Now that we have the data, let's make a component to display each article!
- This is shorthand for a component that only has a “render” function

```
var Article = function({article}) {  
  var imgURL = 'https://static01.nyt.com/' + article.imageURL;  
  return (  
    <div className='article'>  
      <a className='article-link' href={article.webURL}>  
        <img className='article-image'  
          title={article.title}  
          src={imgURL} />  
      </a>  
    </div>  
  );  
}
```

Displaying Data

- Now that we have the data, let's make a component to display each article!
- This is shorthand for a component that only has a “render” function

```
var Article = function({article}) {  
  var imgURL = 'https://static01.nyt.com/' + article.imageURL;  
  return (  
    <div className='article'>  
      <a className='article-link' href={article.webURL}>  
        <img className='article-image'  
          title={article.title}  
          src={imgURL} />  
      </a>  
    </div>  
  );  
}
```

Displaying Data

- Now that we have the data, let's make a component to display each article!
- This is shorthand for a component that only has a “render” function

```
var Article = function({article}) {  
  var imgURL = 'https://static01.nyt.com/' + article.imageURL;  
  return (  
    <div className='article'>  
      <a className='article-link' href={article.webURL}>  
        <img className='article-image'  
          title={article.title}  
          src={imgURL} />  
      </a>  
    </div>  
  );  
}
```


Displaying Data

- Now that we have the data, let's make a component to display each article!
- This is shorthand for a component that only has a “render” function

```
var Article = function({article}) {  
  var imgURL = 'https://static01.nyt.com/' + article.imageURL;  
  return (  
    <div className='article'>  
      <a className='article-link' href={article.webURL}>  
        <img className='article-image'  
          title={article.title}  
          src={imgURL} />  
      </a>  
    </div>  
  );  
}
```

Displaying Data

- Now that we have the data, let's make a component to display each article!
- This is shorthand for a component that only has a “render” function

```
var Article = function({article}) {  
  var imgURL = 'https://static01.nyt.com/' + article.imageURL;  
  return (  
    <div className='article'>  
      <a className='article-link' href={article.webURL}>  
        <img className='article-image'  
          title={article.title}  
          src={imgURL} />  
      </a>  
    </div>  
  );  
}
```

Displaying Data

- Now that we have the data, let's make a component to display each article!
- This is shorthand for a component that only has a “render” function

```
var Article = function({article}) {  
  var imgURL = 'https://static01.nyt.com/' + article.imageURL;  
  return (  
    <div className='article'>  
      <a className='article-link' href={article.webURL}>  
        <img className='article-image'  
          title={article.title}  
          src={imgURL} />  
      </a>  
    </div>  
  );  
}
```

Displaying Data

- Now that we have the data, let's make a component to display each article!
- This is shorthand for a component that only has a “render” function

```
var Article = function({article}) {  
  var imgURL = 'https://static01.nyt.com/' + article.imageURL;  
  return (  
    <div className='article'>  
      <a className='article-link' href={article.webURL}>  
        <img className='article-image'  
          title={article.title}  
          src={imgURL} />  
      </a>  
    </div>  
  );  
}
```

Displaying Data

- Now that we have the data, let's make a component to display each article!
- This is shorthand for a component that only has a “render” function

```
var Article = function({article}) {  
  var imgURL = 'https://static01.nyt.com/' + article.imageURL;  
  return (  
    <div className='article'>  
      <a className='article-link' href={article.webURL}>  
        <img className='article-image'  
          title={article.title}  
          src={imgURL} />  
      </a>  
    </div>  
  );  
}
```

Displaying Data

- Now that we have the data, let's make a component to display each article!
- This is shorthand for a component that only has a “render” function

```
var Article = function({article}) {  
  var imgURL = 'https://static01.nyt.com/' + article.imageURL;  
  return (  
    <div className='article'>  
      <a className='article-link' href={article.webURL}>  
        <img className='article-image'  
          title={article.title}  
          src={imgURL} />  
      </a>  
    </div>  
  );  
}
```

Displaying Data

- Now that we have the data, let's make a component to display each article!
- This is shorthand for a component that only has a “render” function

```
var Article = function({article}) {  
  var imgURL = 'https://static01.nyt.com/' + article.imageURL;  
  return (  
    <div className='article'>  
      <a className='article-link' href={article.webURL}>  
        <img className='article-image'  
          title={article.title}  
          src={imgURL} />  
      </a>  
    </div>  
  );  
}
```

Displaying Data

- Now that we have the data, let's make a component to display each article!
- This is shorthand for a component that only has a “render” function

```
var Article = function({article}) {  
  var imgURL = 'https://static01.nyt.com/' + article.imageURL;  
  return (  
    <div className='article'>  
      <a className='article-link' href={article.webURL}>  
        <img className='article-image'  
          title={article.title}  
          src={imgURL} />  
      </a>  
    </div>  
  );  
}
```


Displaying Data

- Now that we have the data, let's make a component to display each article!
- This is shorthand for a component that only has a “render” function

```
var Article = function({article}) {  
  var imgURL = 'https://static01.nyt.com/' + article.imageURL;  
  return (  
    <div className='article'>  
      <a className='article-link' href={article.webURL}>  
        <img className='article-image'  
          title={article.title}  
          src={imgURL} />  
      </a>  
    </div>  
  );  
}
```

Displaying Data

- Now that we have the data, let's make a component to display each article!
- This is shorthand for a component that only has a “render” function

```
var Article = function({article}) {  
  var imgURL = 'https://static01.nyt.com/' + article.imageURL;  
  return (  
    <div className='article'>  
      <a className='article-link' href={article.webURL}>  
        <img className='article-image'  
          title={article.title}  
          src={imgURL} />  
      </a>  
    </div>  
  );  
}
```

Displaying Data

- Now that we have the data, let's make a component to display each article!
- This is shorthand for a component that only has a “render” function

```
var Article = function({article}) {  
  var imgURL = 'https://static01.nyt.com/' + article.imageURL;  
  return (  
    <div className='article'>  
      <a className='article-link' href={article.webURL}>  
        <img className='article-image'  
          title={article.title}  
          src={imgURL} />  
      </a>  
    </div>  
  );  
}
```

Displaying Data

- Finally, we can render the Articles Grid to tie it all together.

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  render () {  
    return this.state.articles && (  
      <div className='articles'>  
        { this.state.articles.map( function (article) {  
          return <Article article={article} key={article._id} />;  
        }) }  
      </div>  
    );  
  }  
}
```

Displaying Data

- Finally, we can render the Articles Grid to tie it all together.

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  render () {  
    return this.state.articles && (  
      <div className='articles'>  
        { this.state.articles.map( function (article) {  
          return <Article article={article} key={article._id} />;  
        }) }  
      </div>  
    );  
  }  
}
```

Displaying Data

- Finally, we can render the Articles Grid to tie it all together.

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  render () {  
    return this.state.articles && (  
      <div className='articles'>  
        { this.state.articles.map( function (article) {  
          return <Article article={article} key={article._id} />;  
        }) }  
      </div>  
    );  
  }  
}
```

Displaying Data

- Finally, we can render the Articles Grid to tie it all together.

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  render () {  
    return this.state.articles && (  
      <div className='articles'>  
        { this.state.articles.map( function (article) {  
          return <Article article={article} key={article._id} />;  
        }) }  
      </div>  
    );  
  }  
}
```

Displaying Data

- Finally, we can render the Articles Grid to tie it all together.

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  render () {  
    return this.state.articles && (  
      <div className='articles'>  
        { this.state.articles.map( function (article) {  
          return <Article article={article} key={article._id} />;  
        }) }  
      </div>  
    );  
  }  
}
```


Displaying Data

- Finally, we can render the Articles Grid to tie it all together.

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  render () {  
    return this.state.articles && (  
      <div className='articles'>  
        { this.state.articles.map( function (article) {  
          return <Article article={article} key={article._id} />;  
        }) }  
      </div>  
    );  
  }  
}
```

Displaying Data

- Finally, we can render the Articles Grid to tie it all together.

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  render () {  
    return this.state.articles && (  
      <div className='articles'>  
        { this.state.articles.map( function (article) {  
          return <Article article={article} key={article._id} />;  
        }})  
      </div>  
    );  
  }  
}
```

Displaying Data

- Finally, we can render the Articles Grid to tie it all together.

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  render () {  
    return this.state.articles && (  
      <div className='articles'>  
        { this.state.articles.map( function (article) {  
          return <Article article={article} key={article._id} />;  
        }) }  
      </div>  
    );  
  }  
}
```

Displaying Data

- Finally, we can render the Articles Grid to tie it all together.

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  render () {  
    return this.state.articles && (  
      <div className='articles'>  
        { this.state.articles.map( function (article) {  
          return <Article article={article} key={article._id} />;  
        }) }  
      </div>  
    );  
  }  
}
```

Displaying Data

- Finally, we can render the Articles Grid to tie it all together.

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  render () {  
    return this.state.articles && (  
      <div className='articles'>  
        { this.state.articles.map( function (article) {  
          return <Article article={article} key={article._id} />;  
        }) }  
      </div>  
    );  
  }  
}
```

Displaying Data

- Finally, we can render the Articles Grid to tie it all together.

```
class ArticlesGrid extends React.Component {  
  . . .  
  
  render () {  
    return this.state.articles && (  
      <div className='articles'>  
        { this.state.articles.map( function (article) {  
          return <Article article={article} key={article._id} />;  
        }) }  
      </div>  
    );  
  }  
}
```

Summary

- Service Oriented Architecture and Software as a Service (SaaS) simplify the creation of web applications by allowing for combinations of online components
- We can access services via RESTful APIs and develop React components to make use of them