

# parcel-examples (0.0.1)

Maksim Kostromin

Version 0.0.1, 2018-07-01 14:01:19 UTC

# Table of Contents

1. Introduction .....	2
2. Implementation .....	3
2.1. basic example .....	3
2.2. babel stage-0 .....	5
2.3. react .....	5
2.4. preact .....	6
2.5. vue .....	7
2.6. yaml .....	9
2.7. markdown .....	9
2.8. markdowns (2) .....	11
3. Links .....	13

Travis CI status: [Build Status](#)

# Chapter 1. Introduction

Read [reference documentation](#)

other repos with old examples:

- [vue](#)
- [react](#)

generated by [generator-jvm](#) yeoman generator (java)

# Chapter 2. Implementation

## 2.1. basic example

*prepare project*

```
mkdir -p basic

echo "node_modules" >> basic/.gitignore
echo "dist" >> basic/.gitignore
echo ".cache" >> basic/.gitignore
echo '{"scripts":{"start":"parcel src/","build":"parcel build src/index.html"}}' >
basic/package.json

cd basic/
npm init -y
npm i -D parcel-bundler
```

*prepare project files*

```
mkdir -p src
touch src/index.html
touch src/styles.css
touch src/main.js
```

*main.js*

```
(function main() {
  'use strict';
  document.addEventListener('DOMContentLoaded', function onDOMContentLoaded() {
    document.querySelector('#app').innerHTML = '<h1>Hey!    </h1>';
  }, false);
})();
```

*styles.css*

```
html,
body {
  height: 100%;
}
body {
  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', Helvetica, Arial,
  sans-serif, 'Apple Color Emoji', 'Segoe UI Emoji', 'Segoe UI Symbol';
}
#app {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100%;
}
h1 {
  font-weight: 300;
}
```

*index.html*

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport"
    content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Basic | Parcel</title>
  <link rel="shortcut icon" href="./favicon.ico" type="image/x-icon">
  <link rel="stylesheet" href="./styles.css">
</head>
<body>
  <div id="app"></div>
  <script src="./main.js"></script>
</body>
</html>
```

*start development*

```
npm start
```

open [localhost:1234/](http://localhost:1234/)

*build bundle*

```
npm run build
```

verify result

```
npm i -g serve
serve dist/
```

open [localhost:5000/](http://localhost:5000/)

## 2.2. babel stage-0

parcel needed only .babelrc file and installed required presets

add **.babelrc** file, install and configure presets: **env** and **stage-0**

```
cp -Rf basic babel-stage-0
cd babel-stage-0/
echo '{"presets":["env","stage-0"]}' > .babelrc
npm i -D babel-preset-env babel-preset-stage-0
```

now you can use some cool JS features:

```
const obj = { lololo: 'trololo' };

document.querySelector('#app').textContent = JSON.stringify({
  ...obj,
  hey: 'ho!',
});
```

done.

## 2.3. react

for React you need even less: just install react library and parcel will recognize everything for you!

```
cp -Rf babel-stage-0 react
cd react/
npm i -S react react-dom
npm i -D babel-preset-react
echo '{"presets":["env","react"]}' > .babelrc
```

*add react component:*

```
import React, { Component } from 'react';

export class EchoEhlo extends Component {
  constructor() {
    super();
    this.state = {
      message: 'Hey!',
    };
    this.toggle = this.toggle.bind(this);
  }
  toggle() {
    this.setState({
      message: this.state.message.split('').reverse().join(''),
    })
  }
  render() {
    return <h1 onClick={this.toggle}>
      {this.state.message}
    </h1>
  }
}
```

*and update code in `src/main.js` file:*

```
import React from 'react';
import { render } from 'react-dom';
import { EchoEhlo } from './components/echo-ehlo';

render(
  <EchoEhlo/>,
  document.querySelector('#app')
);
```

build, run, verify...

done.

## 2.4. preact



*do necessary updates from react example:*

```
cp -Rf react preact
cd preact/

npm rm react react-dom babel-preset-react

npm i -S preact preact-compat
npm i -D babel-preset-preact babel-plugin-transform-class-properties

echo '{"presets":["env","preact"],"plugins":["transform-class-properties"]}' >
.babelrc
```

*implement preact component:*

```
import { h, Component } from 'preact';

export /* don't works without default 0.0 */ default class EchoEhlo extends Component
{
  state = { message: 'Hey!' };
  toggle = () => {
    const curr = this.state.message;
    const reverced = curr.split('').reverse().join('');
    this.setState({ message: reverced, });
  };
  render({}, { message }, {}) {
    return <h1 onClick={this.toggle}>
      {message}
    </h1>
  }
}
```

*update entry point:*

```
import { h, render } from 'preact';
import HelloEhlo from './components/echo-ehlo';

render(
  <HelloEhlo/>,
  document.querySelector('#app')
);
```

done.

## 2.5. vue

to be able build vue apps, in addition to babel example you only need install vue and two dev packages:

```
cp -Rf babel-stage-0 vue
cd vue/
npm i -S vue
npm i -D babel-preset-vue @vue/component-compiler-utils vue-template-compiler
echo '{"presets":["env","vue"]}' > .babelrc
```

add vue component file: **App.vue**

```
<template lang="html">
  <div id="app">
    <h1>Hey!    </h1>
  </div>
</template>

<script>
  export default {
    name: 'app'
  }
</script>

<style lang="css">
  html,
  body {
    height: 100%;
  }
  body {
    font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', Helvetica, Arial,
    sans-serif, 'Apple Color Emoji', 'Segoe UI Emoji', 'Segoe UI Symbol';
  }
  #app {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100%;
  }
  h1 {
    font-weight: 300;
  }
</style>
```

*main.js*

```
import Vue from 'vue'
import App from './components/App.vue'

new Vue({
  el: '#app',
  render: h => h(App)
});
```

## 2.6. yaml

*to be able to parse YAML we need js-yaml package:*

```
cp -Rf basic yaml
cd yaml/
npm i -ES js-yaml
```

*add YAML file:*

```
app:
  map:
    key: value
  list:
    - value 1
    - value 2
```

*main.js*

```
import app from './app.yaml';

document.addEventListener('DOMContentLoaded', function onDOMContentLoaded() {
  document.querySelector('#app').innerHTML = `
    <h1>Hey!    </h1>
    <pre>${JSON.stringify(app, null, 2)}</pre>
  `;
}, false);
```

## 2.7. markdown

*to be able to parse YAML we need parcel-plugin-markdown package:*

```
cp -Rf basic markdown
cd markdown/
npm i -ED parcel-plugin-markdown
```

add `app.md` file:

```
# app

## ololo

**ololo article**

list:

- one
- two
- three

## trolololololo

  _0_
  |
  / \

## js

<!--

---
title: JavaScript Article'
metadata:
  ololo: trololo
---

-->

### JS article

js, ololo, trololo, javascript, olololo ololo, trololo, olololo ololo, trololo,
olololo ololo,
trololo, olololo, js, ololo, trololo, olololo, js ololo, trololo, olololo.....

```javascript
function ololo(arg) {
  arg = arg || 'trololo';
  console.log('ololo', arg);
}
```
```

main.js

```
import app from './app.md';

document.addEventListener('DOMContentLoaded', function onDOMContentLoaded() {
  document.querySelector('#app').innerHTML = `
    <h1>Hey!    </h1>
    <div>${app}</div>
  `;
}, false);
```



See also [markdowns](#) and [markdowns2](#) projects for non single md-file parsing use case...

## 2.8. markdowns (2)

example how we can handle lazy loading with parcel for markdown files lazy processing on runtime

*add ./src/posts folder with some posts in markdown format:*

```
tree markdowns2/src/posts/
markdowns2/src/posts/
├── 2018
│   └── 06
│       ├── 29
│       │   ├── 01-ololo.md
│       │   └── 02-trololo.md
│       └── 30
│           ├── 01-javascript.md
│           └── 02-bash.md
```

*main.js implementation:*

```
import marked from 'marked';

const years = require('./posts/**/*.*md');

Object.keys(years).map(year => {
  const months = years[year];
  Object.keys(months).map(month => {
    const days = months[month];
    Object.keys(days).map(day => {
      const posts = days[day];
      Object.keys(posts)
        .map(src => posts[src])
        .forEach(uri => fetch(uri)
          .then(resp => resp.text())
          .then(markdown => marked(markdown))
          .then(html => document.querySelector('#app').innerHTML += html));
    });
  });
});
```

# Chapter 3. Links

- [parcel](#)
- [parcel examples](#)
- [react](#)
- [preact](#)
- [vue](#)

- 
- [GitHub repo](#)
  - [GitHub pages](#)