Original Series

The streaming build system

https://gulpjs.com/

GROUND UP SERIES

GULP.JS

Syed Awase Khirni

Syed Awase earned his PhD from University of Zurich in GIS, supported by EU V Framework Scholarship from SPIRIT Project (www.geo-spirit.org). He currently provides consulting services through his startup www.territorialprescience.com and www.sycliq.com . He empowers the ecosystem by sharing his technical skills worldwide, since 2008. He provides training in Java Technology Stack, .Net Technology Stack, R, DataScience, Client Side frameworks (Angular,KnockOut,Aurelia,Vue,Ember, Backbone etc..), Node Stack, Machine Learning, Python Stack, Php Stack.

www.territorialprescience.com

www.sycliq.com

Terms of Use

You shall not circulate these slides without written permission from **Territorial Prescience Research I Pvt ltd.**

If you use any material, graphics or code or notes from these slides, you shall seek written permission from TPRI and acknowledge the author Dr. Syed Awase Khirni

If you have not received this material, post-training session, you shall destroy it immediately and not use it for unauthorized usage of the material. If any of the material, that has been shared is further used for any unauthorized training by the recipient, he shall be liable to be prosecuted for the damages. Any supporting material that has been provided by the author, shall not be use directly or indirectly without permission.

If this material, has been shared to any organization prior to the training and the organization does not award the contract to TPRI, it should not use the training material internally. If by any chance, the organization is using this training material without written permission, the organization is liable to pay for the damages to TPRI and is subjected to legal action, jurisdiction being Bangalore. It shall also pay for any expenses, legal, recovery and all applicable damages and costs incurred by TPRI.

TPRI has right to claim damages ranging from USD 50000 to USD 10,0000 dollars as damages, for unauthorized usage.

Any organization, which does not intend to go ahead with training or does not agree with the terms and conditions, should destroy the material from its network immediately. The burden of proof lies on the client, with whom this material has been shared.

Recovery of the damages and all expenses incurred including legal fees will be born by the client organization/candidate/party, which has violated these terms and conditions.

Only candidates who have attended the training session in person from Dr. Syed Awase Khirni, TPRI are entitled to hold this training material. They cannot further circulate it, or use it or morph it, or change it to provide trainings. This training material cannot be used by any other candidates other than the registered individuals for the class room based session.

TPRI reserves all the rights to this material and code plays and right to modify them as and when it deems fit.

If you agree with the terms and conditions, please go ahead with using the training material. Else please close and destroy the slide and inform TPRI immediately.

Slide Version Updates

Last Updated	Version	Release Date	Updated by	Code Plays Done @

Open Source

Continuous integration and deployment

Gulp:Introduction

SYED AWASE KHIRNI

Gulp

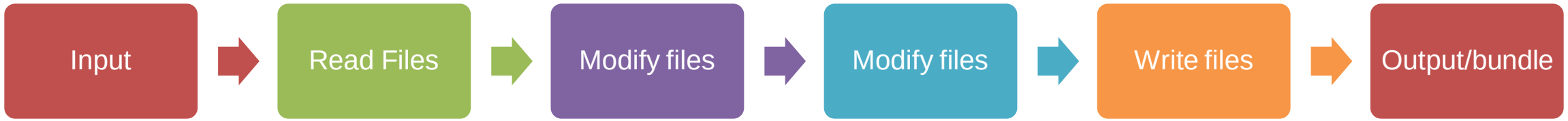
- A streaming javascript build system built with node.js that leverages the power of streams and code-over-configuration to automate, organize and run development tasks
- Configured with code instead of using configuration files
- A streaming build system
- Allows you to modify, process project files and chain/pipe tasks
- Built on top of node.js and npm
- An advanced task manager for automating javascript development .
- Must have **node** and **npm** installed.

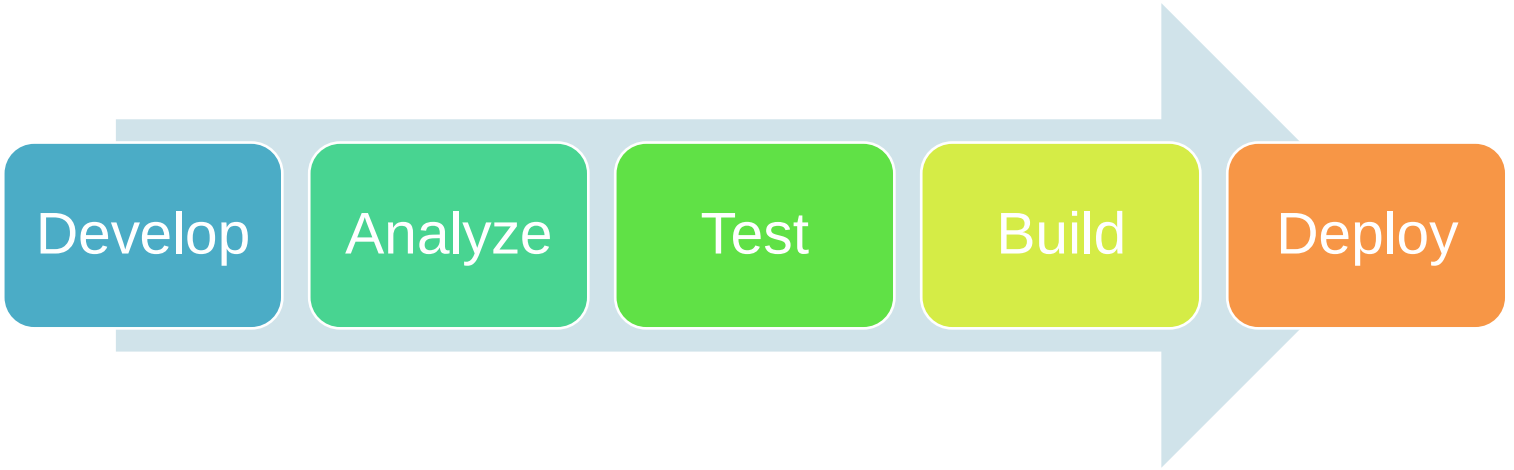
Why build system?

- It can do the repetitive things that have to be done constantly and consistently

- What Build Systems do
  - File Concatenation
  - CSS pre-processing from LESS/SASS
  - Transpiling code from ES6/Typescript.
  - Static Code Analysis using JSHINT/JSLINT, code compliance
  - Run local dev server for live-reloading the changes made.
  - Live browser re-load/rendering with browser
  - Provides “always ready” builds for rendering for continuous integration and deployment.

Gulp build workflow





Resource bundle process for production

- Minify
- concatenate
- Vendor prefixes
- Less to CSS compilation
- Vendor and custom code
- Injecting files into html
- Cache busting
- Framework template cache
- Testing
- Code analysis

Build Systems

GULP | BROWSERIFY | YEOMAN | GRUNT | BROCCOLI | WEBPACK | BRUNCH

Name		No of Plugins	Task Instruction Style	Processing method	Last updated
Gulp	Stream Paradigm	1100+	code	Pipeline	10,oct 2017
NPM		10000+	Configuration	Pipeline	9,Nov 2017
Grunt	Linear fashion	4000+	Configuration	File-based	18,Aug,2017
Brunch		~100	Code	Pipeline	25 Aug 2017
Broccoli		200+	Configuration	pipeline	17 May,2017
Fly		5	Code	Pipeline	5 April,2016
Gobble		50	Configuration	File-based	15,Jul,2015
Yeoman					
Webpack	Module bundler		configuration		9 Nov,2017

Build Systems

pundle

https://github.com/steelbrain/pundle

https://github.com/rollup/rollup

rollup.js

Next-generation ES6 module bundler

https://stealjs.com/

http://fuse-box.org/

assetgraph

https://github.com/assetgraph/assetgraph

https://github.com/vigetlabs/blendid

Original Series

Gulp

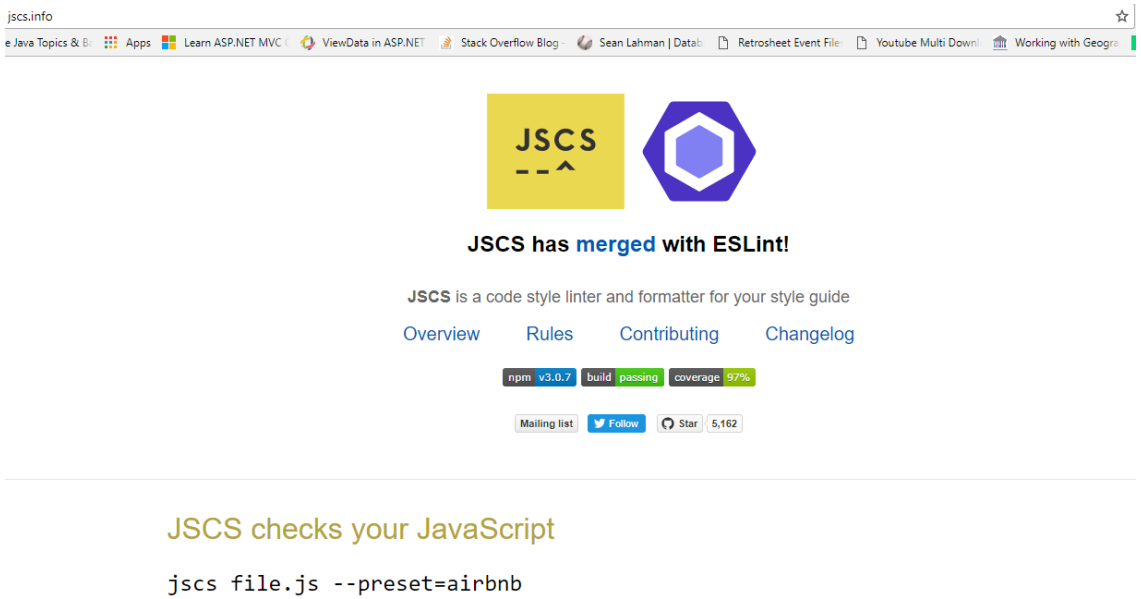
- Code over configuration
- Stream based
- 1100+ plugins
- Use node more readily
- Easier to debug
- Easier to read and write

Grunt

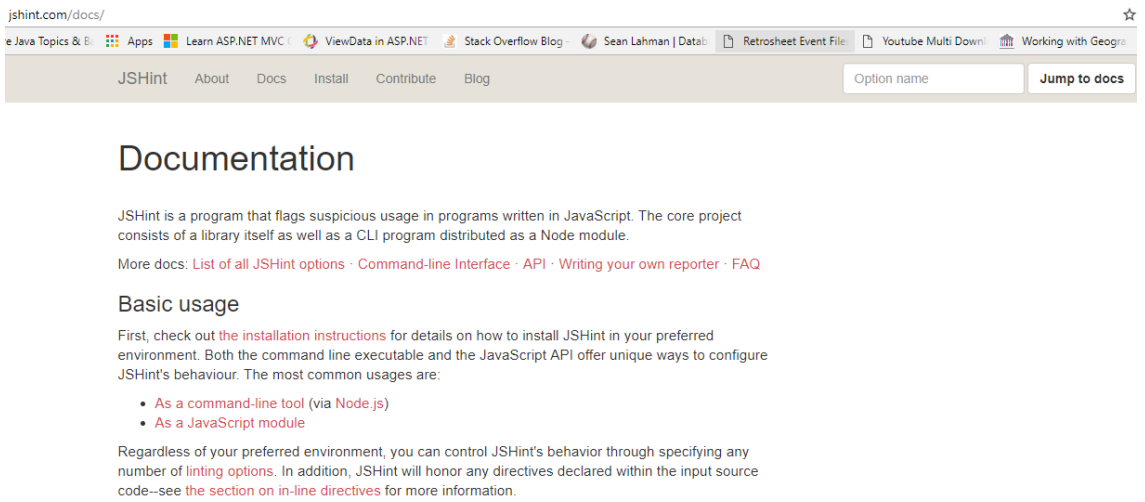
- Configuration over code
- File based
- 3900+plugins

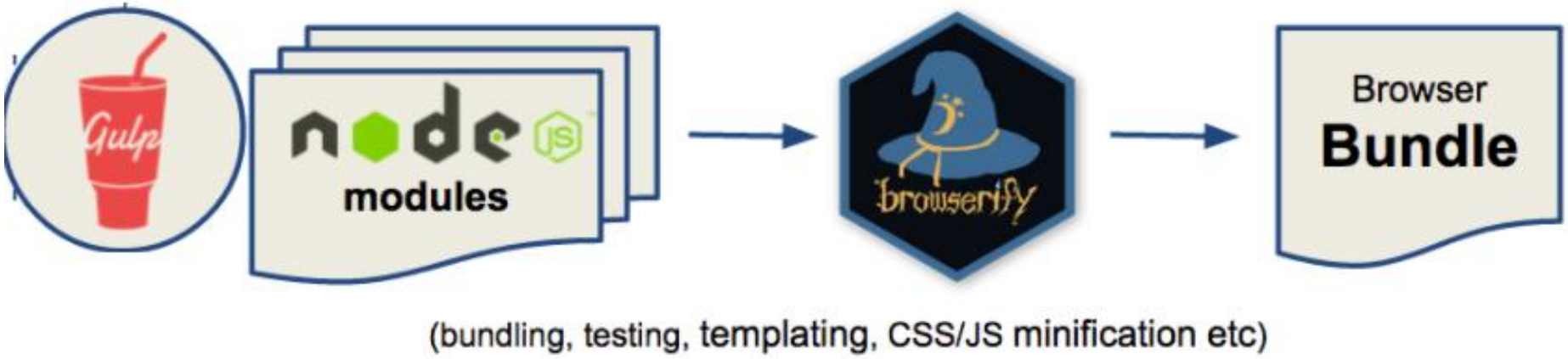
Hinting and Linting

http://jscs.info/



http://jshint.com/docs/





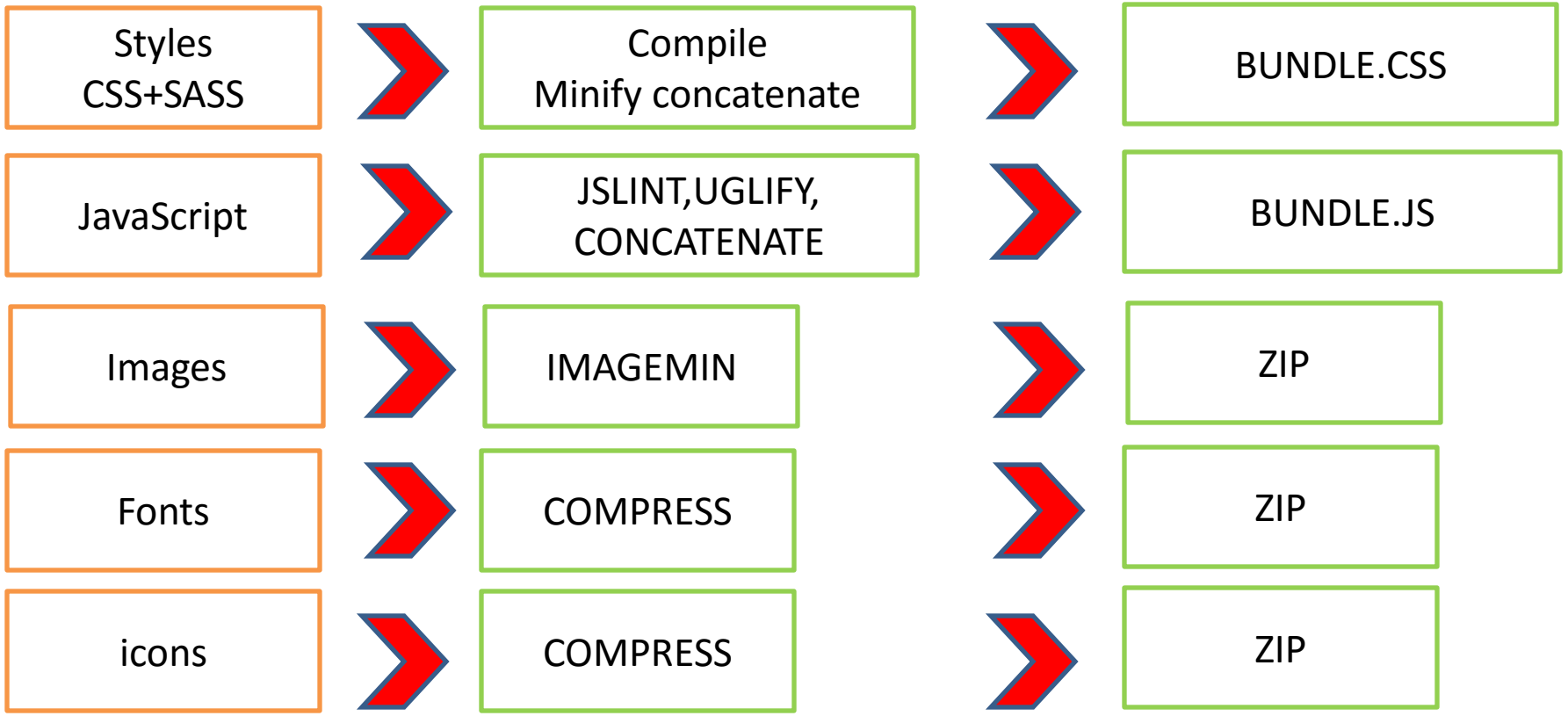
SYED AWASE KHIRNI

GULP: SETTING UP!

Build Process

COMPILE CODE → PACKAGE FOR DEPLOYMENT

Modules with dependencies



Included in Index.html

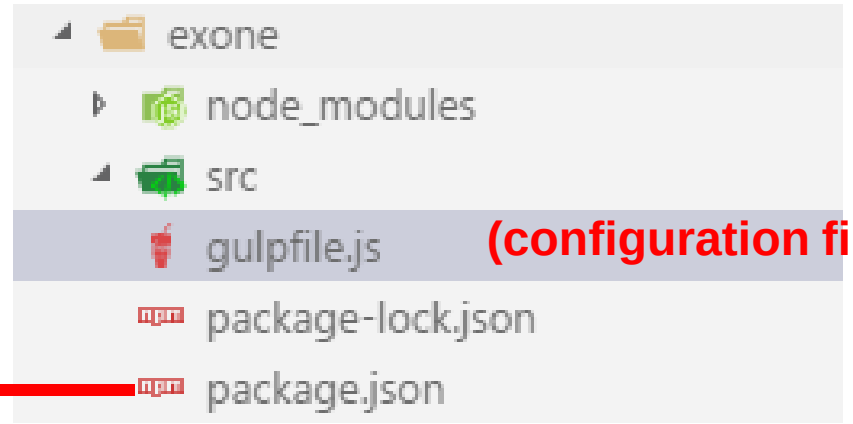
Bundle.zip

Gulp:Setup

```
// installing gulp
npm init -y //=> to create a default package.json
//installing gulp and gulp cli
npm install --save gulp gulp-cli
```

Package.json

```
{
  "name": "gulpworkflowdemo",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "gulp": "^3.9.1",
    "gulp-cli": "^1.4.0"
  }
}
```



Gulp installation

1. • Install gulp globally

   npm install –g gulp gulp-cli

2. • Install dependent packages
3. • Codify gulp tasks
4. • Execute the task

SYED AWASE KHIRNI

GULP: MANAGING PROJECTS WITH TASKS

Higher level functions

Gulp.task()

- gulp.task()
  - Is used to define a task.
  - A task is a single unit of work

Gulp.src()

- Is used to define the location of files to be processed.
- We use relative path references and wild card syntax to identify specific file types.

Gulp.dest()

- Location where generated files should be written to(destination)
- Used at the end of the chain process

Gulp.watch()

- Instruct gulp to watch a particular file for changes or a directory to automate tasks subsequently
- Session in which tasks are automatically performed.

Gulp.task()

- a higherlevel function
- Registers a task name with a function, optionally declare dependencies.
- **Gulp.task(name,[,dep],fn)**
  - Registers a task name with a function
  - Optionally declare dependencies.
  - Note that '**jshint**' task runs before, task '**js**' is executed.
    - Dependency tasks run in parallel, not in sequence.

```
gulp.task('js',function(){
    return gulp
        .src('./src/**/**/*.js')
        .pipe(concat('vendorall.js'))
        .pipe(uglify())
        .pipe(gulp.dest('./prod/build'));
});
```

```
gulp.task('js',['jshint'],function(){
    return gulp
        .src('./src/**/**/*.js')
        .pipe(concat('vendorall.js'))
        .pipe(uglify())
        .pipe(gulp.dest('./prod/build'));
});
```

Gulp.src/Gulp.src(glob[,options])

- Takes a file system glob(set of files)
- Emits files that match
- Optionally specify options to apply to the glob
- **Options.base**
  - Defines how much of the path to retain.
  - Defaults to everything before the glob

```
gulp.task('js',['jshint'],function(){
    return gulp
        .src('./src/**/**/*.js')
        .pipe(concat('vendorall.js'))
        .pipe(uglify())
        .pipe(gulp.dest('./prod/build'));
});
```

```
gulp.task('js',['jshint'],function(){
    return gulp
        .src('./src/**/**/*.js',{base:'./src/'})
        .pipe(concat('vendorall.js'))
        .pipe(uglify())
        .pipe(gulp.dest('./prod/build'));
});
```

Gulp.dest/gulp.dest(folder[,options])

- Piped files are written to the file system.
- Optionally specify options to apply to the output folder

```
gulp.task('js',['jshint'],function(){
    return gulp
        .src('./src/**/**/*.js',{base:'./src/'})
        .pipe(concat('vendorall.js'))
        .pipe(uglify())
        .pipe(gulp.dest('./prod/build'));
});
```

Gulp.watch/gulp.watch(glob[,options],tasks)

- Run one or more tasks when a file matched by the glob changes
- Options => takes in an array of task names to watch
- Callback is passed event object with type and path.

```
gulp.task('jslint-watch',function(){
    gulp.watch('./src/**/*.js',[
        'jshint'
    ]);
});
```

```
gulp.task('jslint-watch', function () {
    gulp.watch('./src/**/*.sass', function(event){
        console.log('watched event', + event.type+
        'for'+event.path);
    });
});
```

Example one

Consolelog task

- create task to log in the console recursively.

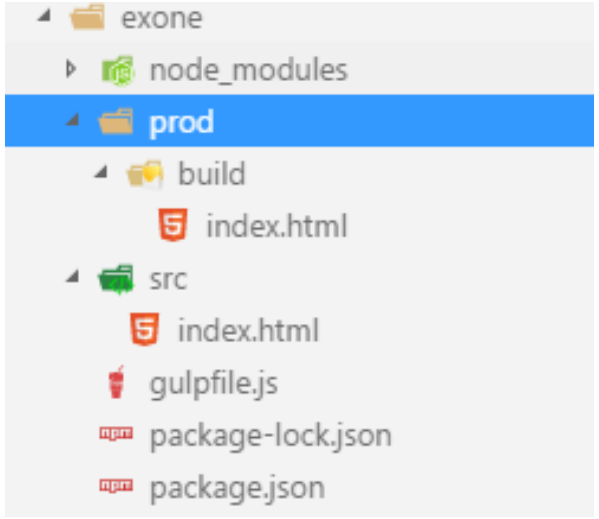
```
//configuration file
var gulp = require('gulp');

gulp.task('logtask',function(){
    for( i=0;i<10;i++){
    console.log("we start to write to the logger");
    }
});
```

```
PS E:\CT\NodeJS\Dec2017NodejsDanskePlan\gulpworkflowdemo\exone> gulp logtask
[20:59:18] Using gulpfile E:\CT\NodeJS\Dec2017NodejsDanskePlan\gulpworkflowdemo\exone\gulpfile.js
[20:59:18] Starting 'logtask'...
we start to write to the logger
we start to write to the logger
we start to write to the logger
we start to write to the logger
we start to write to the logger
we start to write to the logger
```

Example two

- Copy all the html files from source to production
  - gulp



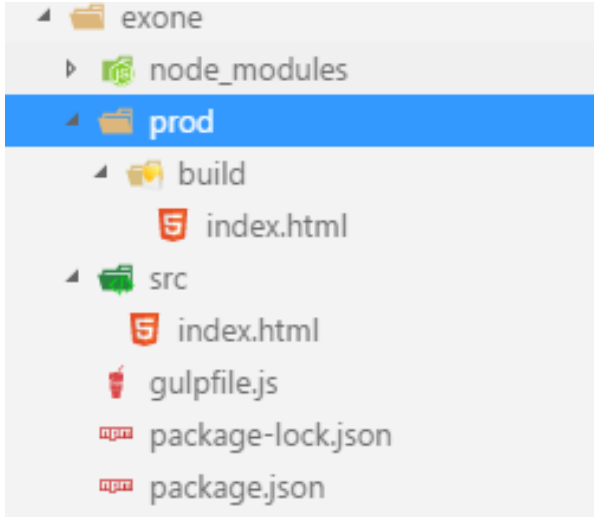

```
//configuration file
var gulp = require('gulp');

gulp.task('logtask',function(){
    for( i=0;i<10;i++){
    console.log("we start to write to the logger");
    }
});
//copy all the htmlfiles from source to production build
gulp.task('copyAllHtmlFiles', function(){
    gulp.src('src/*.html')
        .pipe(gulp.dest('prod/build'));
});

//default task => run without any arguments
gulp.task('default', ['logtask', 'copyAllHtmlFiles']);
```

Example three

- Copy all the html files from source to production
  - gulp




```
//configuration file
var gulp = require('gulp');

gulp.task('logtask',function(){
    for( i=0;i<10;i++){
    console.log("we start to write to the logger");
    }
});
//copy all the htmlfiles from source to production build
gulp.task('copyAllHtmlFiles', function(){
    gulp.src('src/*.html')
        .pipe(gulp.dest('prod/build'));
});

//default task => run without any arguments
gulp.task('default', ['logtask', 'copyAllHtmlFiles']);
```

Installing Gulp plug-in

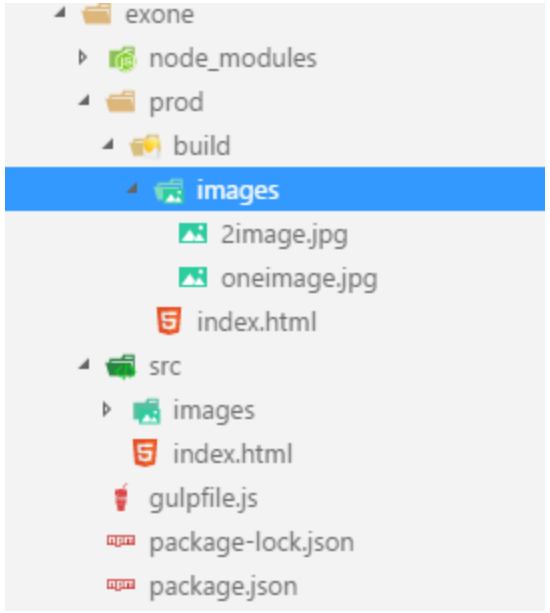
- gulp has a number of plugins that enable us to achieve a specific functionality, by including them.

```
//installing gulp hint and lint plugins
npm install --save gulp-jshint
npm install --save gulp-jscs
// load gulp plugins
npm install --save yargs gulp-load-plugins gulp-if
gulp-print jshint-stylish gulp-util
//image minifier
npm install --save gulp-imagemin
```

https://github.com/sindresorhus/gulp-imagemin

Gulp task to minify images

- minifying the images using gulp-imagemin

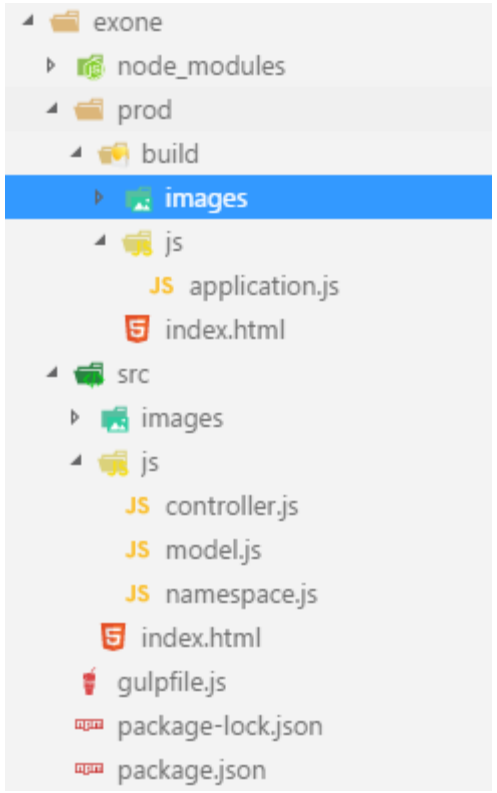



```
//configuration file
var gulp = require('gulp');
var imagemin = require('gulp-imagemin');
gulp.task('logtask',function(){
    for( i=0;i<10;i++){
    console.log("we start to write to the logger");
    }
});
//copy all the htmlfiles from source to production build
gulp.task('copyAllHtmlFiles', function(){
    gulp.src('src/*.html')
        .pipe(gulp.dest('prod/build'));
});
//task to minify images.
gulp.task('minifyImages', function(){
    gulp.src('src/images/**/*')
    .pipe(imagemin())
    .pipe(gulp.dest('prod/build/images'));
})

//default task => run without any arguments
gulp.task('default', ['logtask', 'copyAllHtmlFiles', 'minifyImages']);
```

Uglify and Concat Plugin

- npm install --save gulp-uglify gulp-concat

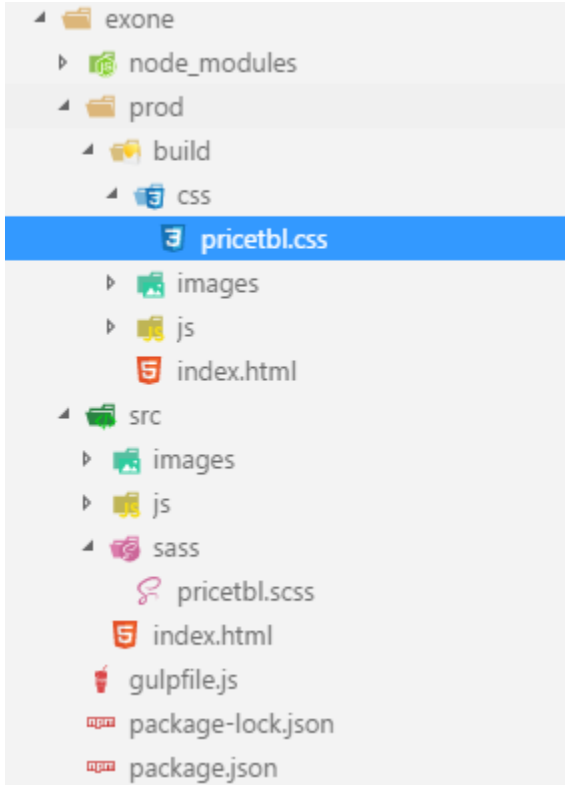



```
//configuration file
var gulp = require('gulp');
var imagemin = require('gulp-imagemin');
var uglify=require('gulp-uglify');
var concat = require('gulp-concat');
gulp.task('logtask',function(){
    for( i=0;i<10;i++){
    console.log("we start to write to the logger");
    }
});
//copy all the htmlfiles from source to production build
gulp.task('copyAllHtmlFiles', function(){
    gulp.src('src/*.html')
        .pipe(gulp.dest('prod/build'));
});
//task to minify images.
gulp.task('minifyImages', function(){
    gulp.src('src/images/**/*')
    .pipe(imagemin())
    .pipe(gulp.dest('prod/build/images'));
});
//minify and concat js file
gulp.task('minifyjs',function(){
    gulp.src('src/js/**/*.js')
        .pipe(uglify())
        .pipe(concat(app.js))
        .pipe(gulp.dest('prod/build/js'));
});

//default task => run without any arguments
gulp.task('default', ['logtask', 'copyAllHtmlFiles', 'minifyImages','minifyjs']);
```

Gulp-Sass plugin

- Npm install –save gulp-sass



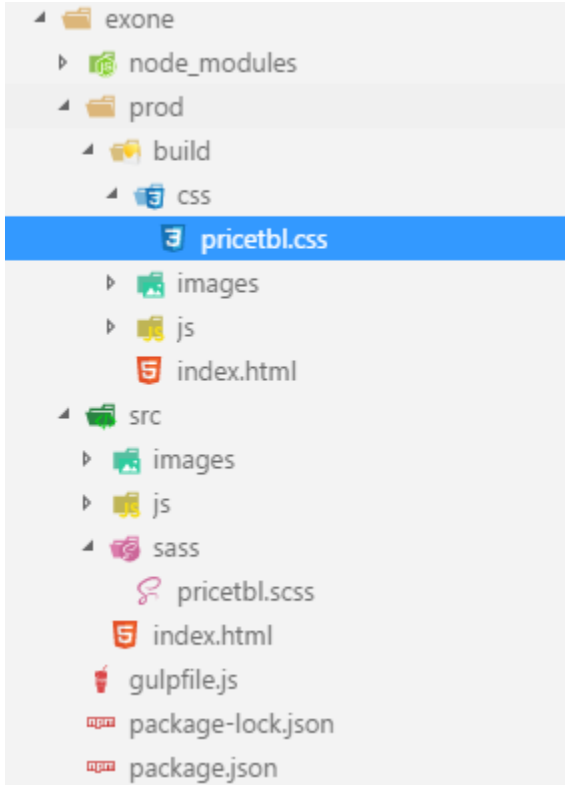

```
var sass =require('gulp-sass');
```

```
//sass to css conversion
gulp.task('sassconverter',function(){
    gulp.src('src/sass/**/*.scss')
        .pipe(sass())
        .pipe(gulp.dest('prod/build/css'));
});
```

```
//default task => run without any arguments
gulp.task('default', ['logtask', 'copyAllHtmlFiles', 'minifyImages','minifyjs', 'sassconverter']);
```

Gulp-Sass plugin

- Npm install –save gulp-sass




```
var sass =require('gulp-sass');
```

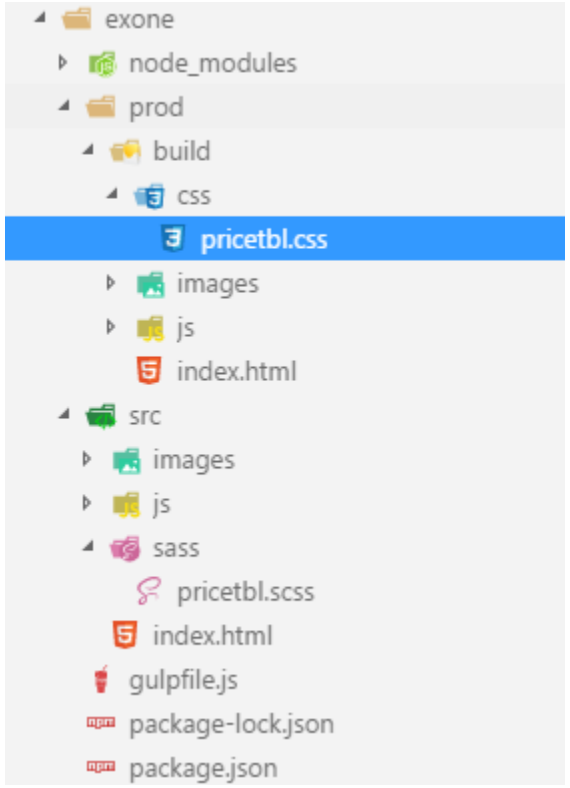
Logging error during conversion of SASS

```
//sass to css conversion
gulp.task('sassconverter',function(){
    gulp.src('src/sass/**/*.scss')
        .pipe(sass().on('error',sass.logError))
        .pipe(gulp.dest('prod/build/css'));
});
```

```
//default task => run without any arguments
gulp.task('default', ['logtask', 'copyAllHtmlFiles', 'minifyImages','minifyjs', 'sassconverter']);
```

Gulp-Sass plugin

- Npm install –save gulp-sass




```
var sass =require('gulp-sass');
```

Output Style Compressed

```
//sass to css conversion
gulp.task('sassconverter',function(){
    gulp.src('src/sass/**/*.scss')
        .pipe(sass({outputStyle:'compressed'}).on
        ('error',sass.logError))
        .pipe(gulp.dest('prod/build/css'));
});
```

```
//default task => run without any arguments
gulp.task('default', ['logtask', 'copyAllHtmlFiles', 'minifyImages','minifyjs', 'sassconverter']);
```

Gulp Task Summary

Gulp.task : define task **Gulp.task('name',function(){})**

1. Gulp.src : read files
2. Gulp.pipe : concatenate
3. Gulp.dest :write files
4. Gulp.watch :watch files

Speed, Efficiency, Continuous Integration, Change Management

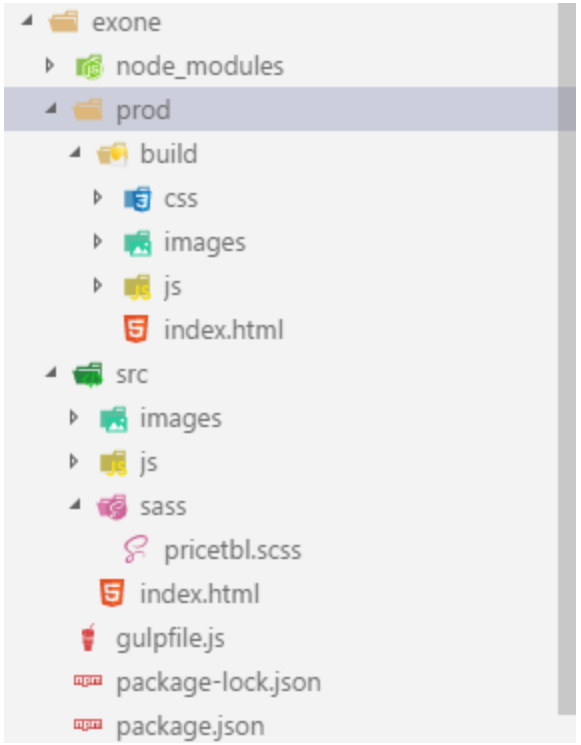
SYED AWASE KHIRNI

GULP: FOR DEV ENVIRONMENT

Run-sequence plugin

```
var runSequence = require('run-sequence');
```

- Runs a sequence of gulp tasks in a specific order.
- Used when a defined run-order, but choose not to or cannot use dependencies.
- Npm install –save run-sequence




```
gulp.task('rsBuild', function(callback){
    runSequence(['logtask', 'copyAllHtmlFiles', 'minifyImages', 'minifyjs', 'sassconverter'],callback);
});
```

Concat-css

- Concatenates css files, bubbling up @import statements (as per the **standard**), and optionally rebasing urls and inlining local @import statements.
- Npm install –save gulp-concat-css

```
var concatCss = require('gulp-concat-css');
```

```
gulp.task('bundleCss', ['sassconverter'],function () {
    return gulp.src('src/css/**/*.css')
        .pipe(concatCss("bundle.css"))
        .pipe(gulp.dest('prod/build/css'));
});
```

```
gulp.task('rsBuild', function(callback){
    runSequence(['logtask', 'copyAllHtmlFiles', 'minifyImages', 'minifyjs', 'sassconverter',
    'bundleCss'],callback);
});
```

Gulp-less to css conversion

- Gulp-less : compile less to css
- Gulp-auto-prefixer : add vendor prefixes
- On('error',function): to handle events.
- Gulp-plumber: gracefully handle errors in watches
- Callbacks : end the stream.

Gulp.watch()

```
//watching for changes
gulp.watch('file-to-watchfor',['tasks','to','execute']);
```

- Gulp.watch is used to monitor source files. It is triggered when any changes to the source file are made, the watch will run an appropriate task.

```
gulp.task('watch',function(){
    gulp.watch('src/**/*.html', ['copyAllHtmlFiles']);
    gulp.watch('src/js/**/*.js',['minifyjs']);
    gulp.watch('src/images/**/*',['minifyImages']);
    gulp.watch('src/sass/**/*.scss',['sassconverter']);
    gulp.watch('src/css/**/*.css',['bundleCss']);
});
```

Gulp Live Releoad

- It specifies the changes in the file system. BrowserSync is used to watch all HTML and CSS files in the CSS directory and perform live reload to that needs to be rendered in the browser.
- BrowserSync makes the workflow faster by synchronizing URLs, interactions, and code changes across multiple devices.

```
var browsersync = require('browser-sync').create();

gulp.task('watch',['browsersyncer'],function(){
    gulp.watch('src/**/*.html', ['copyAllHtmlFiles']);
    gulp.watch('src/js/**/*.js',['minifyjs']);
    gulp.watch('src/images/**/*',['minifyImages']);
    gulp.watch('src/sass/**/*.scss',['sassconverter']);
    gulp.watch('src/css/**/*.css',['bundleCss']);
    //reloading
    gulp.watch('src/**/*.html', browsersync.reload);
    gulp.watch('src/js/**/*.js', browsersync.reload);
    gulp.watch('src/images/**/*', browsersync.reload);
    gulp.watch('src/sass/**/*.scss', browsersync.reload);
    gulp.watch('src/css/**/*.css', browsersync.reload);
});
//browserSync
gulp.task('browsersyncer',function(){
    browsersync.init({
        server:'./src',
        port:'8888',
        ui:{port:8086}
    });
});
```

fully automated build, bundled, compressed distribution, deployment

SYED AWASE KHIRNI

GULP: FOR PRODUCTION ENVIRONMENT

Gulp-useref

```
//installing useref
npm install --save gulp-useref
```

```
var useref = require('gulp-useref');

gulp.task('production', function(){
    return gulp.src('src/**/*.html')
        .pipe(useref())
        .pipe(gulp.dest('prod/build'));
});
```

- It is used to parse build blocks in HTML files to replace references to non-optimized scripts.
- Optimizes all HTML and CSS

```
<!-- build:js  js/mvcapp.min.js-->
<script type="text/javascript" src="./js/namespace.js"></script>
<script type="text/javascript" src="./js/model.js"></script>
<script type="text/javascript" src="./js/controller.js"></script>
<!-- endbuild-->
```

Gulp-useref with uglify

- gulpIf
- Npm install –save gulp-if

```
var gulpIf = require('gulp-if');

//production and minification
gulp.task('prodminify',function(){
    return gulp.src('src/**/*.html')
        .pipe(useref())
        .pipe(gulpIf('*.js',ugilify()))
        .pipe(gulp.dest('prod/build'));
});
```

Preprocess SASS files for Production

```
//deployment build
gulp.task('deploy',function(callback){
    runSequence('sassconverter',['minifyImages','bundleCss','prodminify'],callback);
});
```

Debugging in Production using sourcemaps

https://developers.google.com/web/tools/chrome-devtools/?utm_source=dcc&utm_medium=redirect&utm_campaign=2016q3#source-maps

- Source maps provide the ability to debug in a live environment.
- It is used to run minified javascript (which is not particularly readable in a debugger by itself)
- It gives a readable form of your source for debugging when loaded in browser
- A source map provides a way of mapping code within a compressed file back to it's original position in a source file.
- Developer tools can parse sourcemaps to show original file details
- Sourcemaps can be included in the gulp workflow using
  - Gulp-sourcemaps
  - Lazypipe
- Npm install –save gulp-sourcemaps lazypipe

Sourcemaps, Lazypipe (JS)

```
var sourcemaps = require('gulp-sourcemaps');
var lazypipe = require('lazypipe');

//production and minification
gulp.task('prodminify',function(){
    return gulp.src('src/**/*.html')
        .pipe(useref({},lazypipe().pipe(sourcemaps.init,{loadMaps:true})))
        .pipe(sourcemaps.write('maps'))
        .pipe(gulpIf('*.js',ugilify()))
        .pipe(gulp.dest('prod/build'));
});
```

This is added to js file:

//# sourceMappingURL=/path/to/script.js.map

Sourcemaps,Lazypipe (CSS)

```
<!--build:css css/styles.min.css-->
<link rel="stylesheet" href="style.css">
<!--endbuild-->
```

```
//deployment build
gulp.task('deploy',function(callback){
    runSequence('sassconverter', 'bundleCss',['minifyImages','prodminify'],callback);
});
```

Executed in parallel

Configure BrowserSync to production folder

```
//browserSync
gulp.task('browsersyncer',function(){
    browsersync.init({
        server:'./prod/build/',
        port:'8888',
        ui:{port:8086}
    });
});
```

Gulp Version 4

Gulp.series

- Gulp.series(fn,[,fn,…n])
- Identifies a set of tasks to run in sequence
- Accepts set of functions or task(strings)

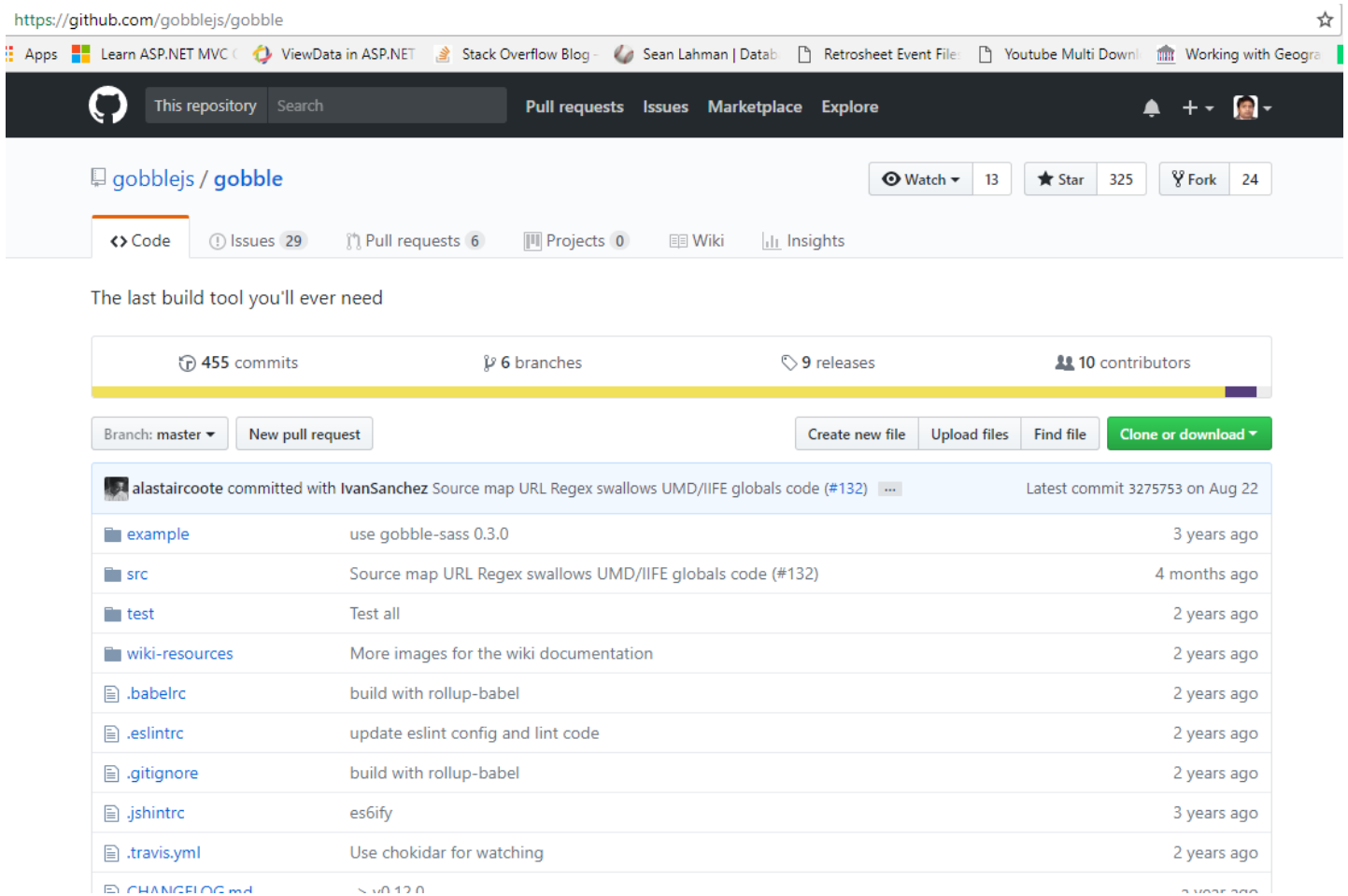
Gulp.parallel

- Gulp.parallel(fn,[,fn,…n])
- Identifies a set of tasks to run in parallel
- Accepts set of functions or task(strings)

OTHER RESOURCES

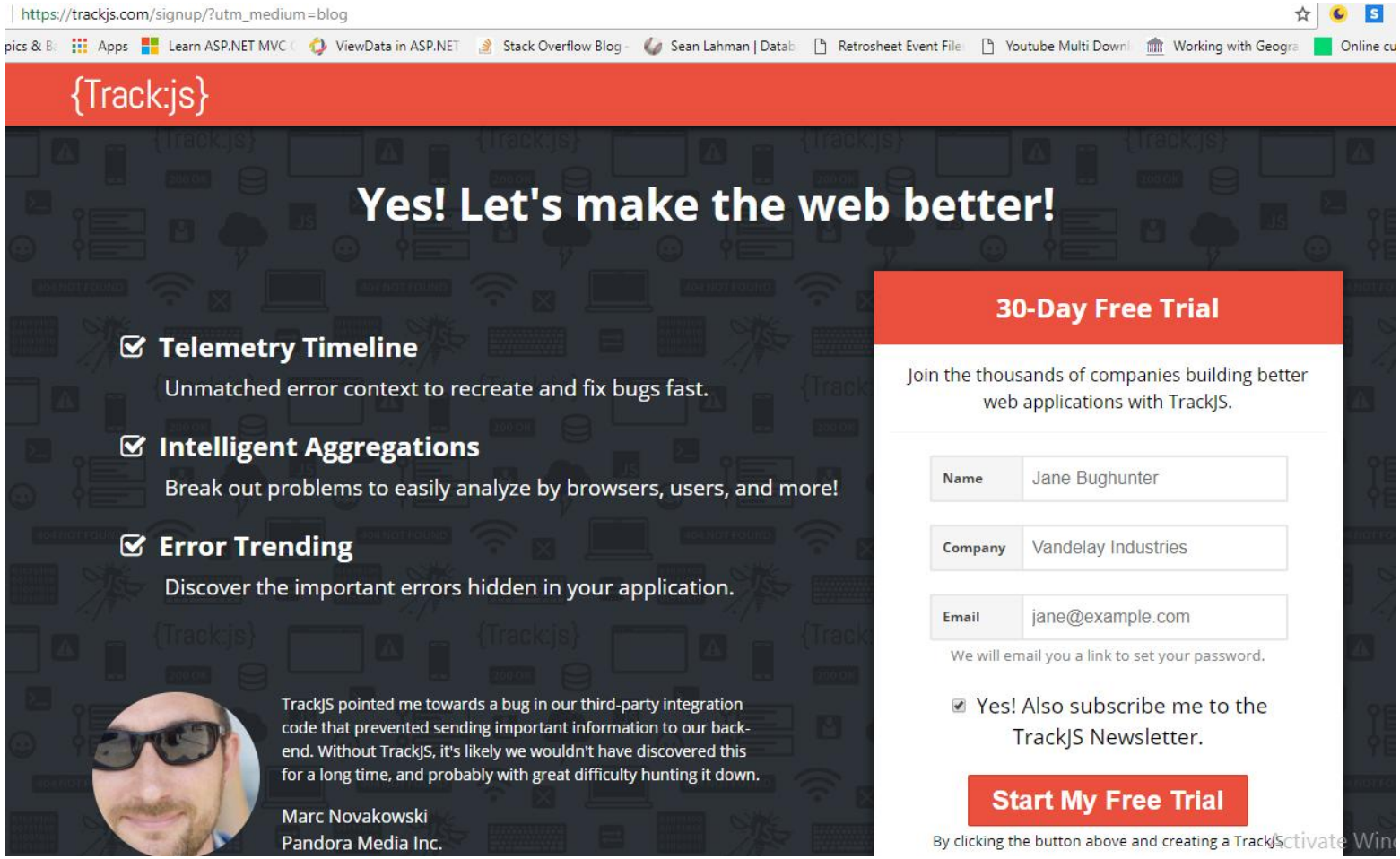
Gobble

https://github.com/gobblejs/gobble



Track.js

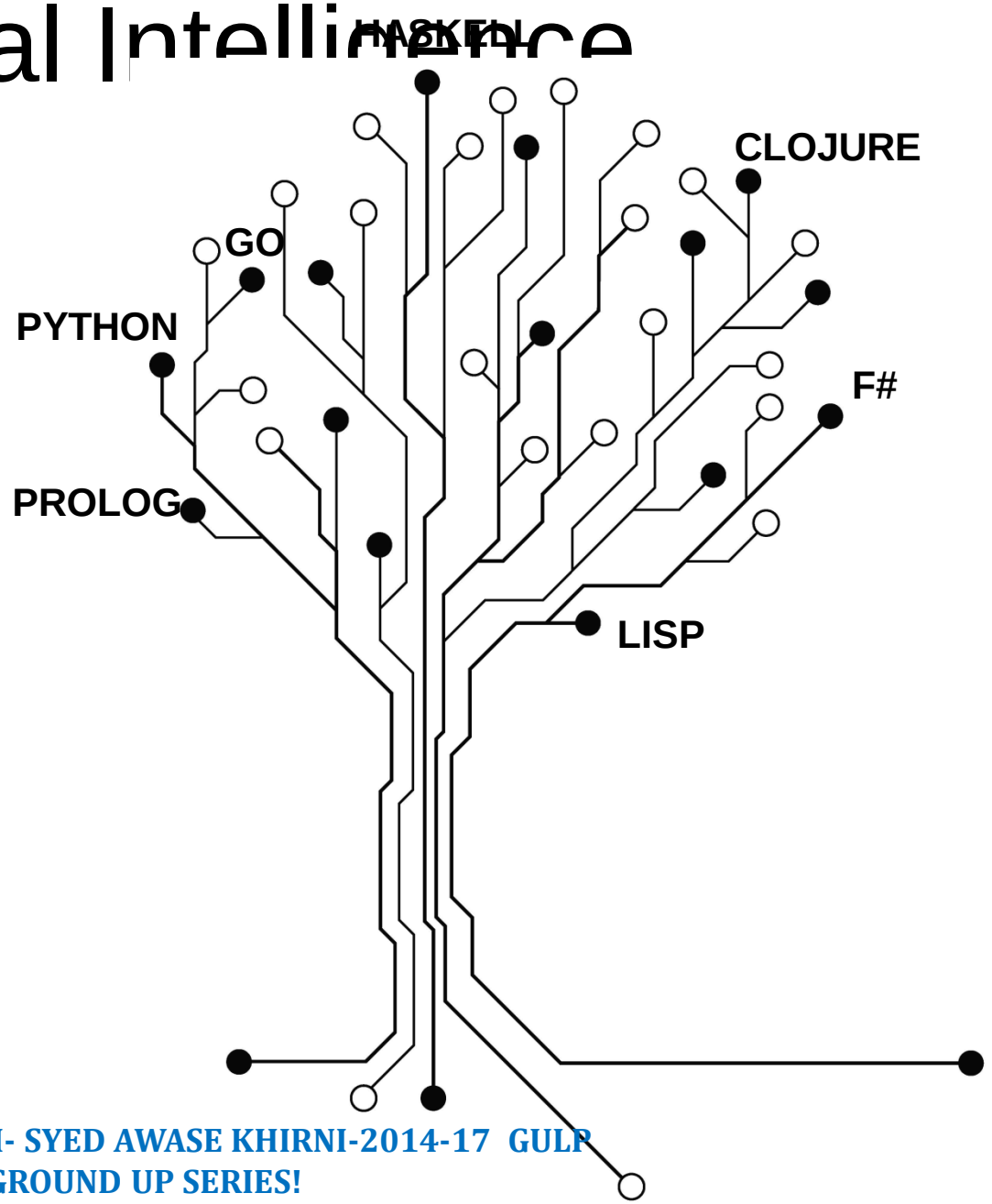
https://trackjs.com/signup/?utm_medium=blog



Empowering You

TPRI-SYCLIQ PROGRAMS OVERVIEW

Artificial Intelligence



We also train on AI Stack

Reach out to us sak@sycliq.com or sak@territorialprescience.com

www.territorialprescience.com

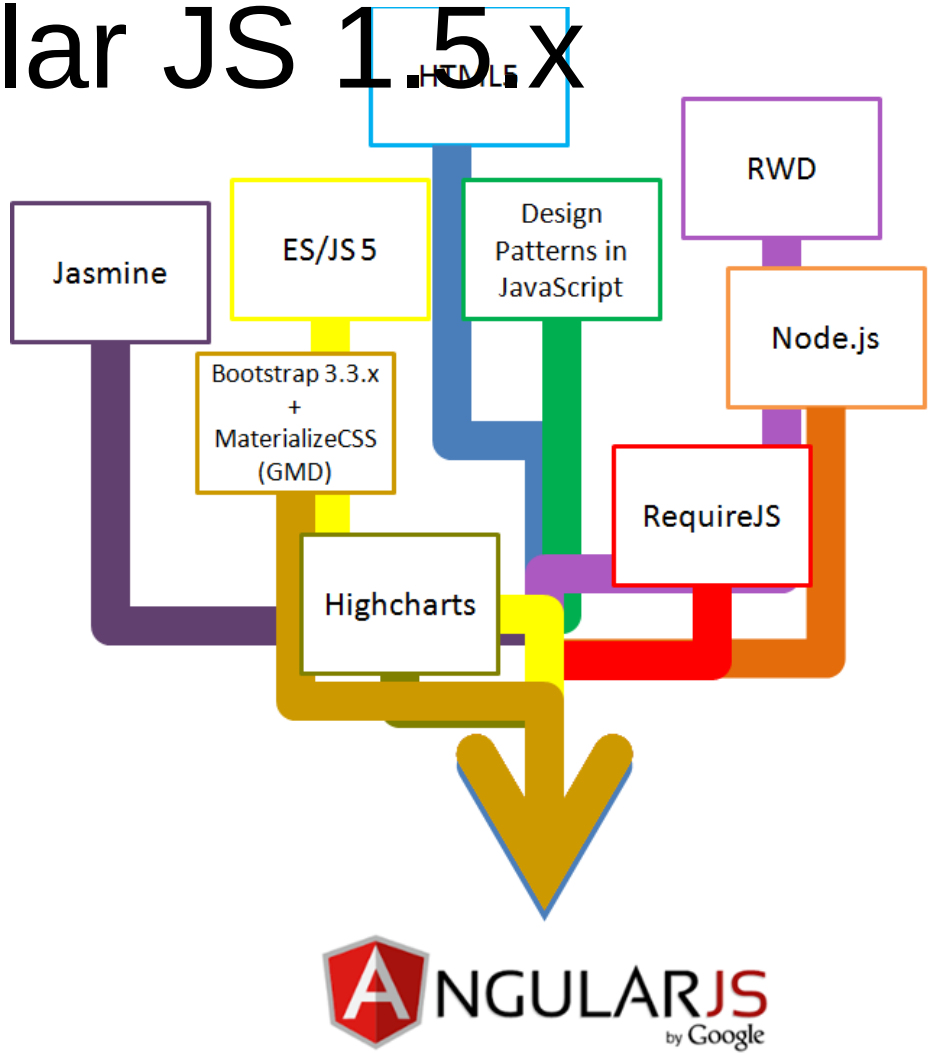
www.sycliq.com

+91.9035433124

Angular 4.x/Angular JS 1.5.x

Dr. Syed Awase 2016 Session Feedbacks: http://bit.ly/2hhNg58

Reach out to sak@territorialprescience.com/+91.9035433124



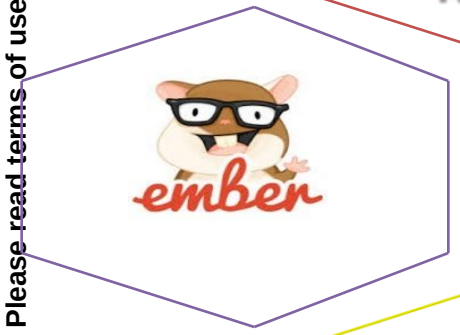
Dr. Syed Awase also offers Machine learning Stack, R Statistical Stack, .NET Stack, Java Stack, RaspberryPi Stack. Get the pulse of performance from here http://bit.ly/2hhNg58

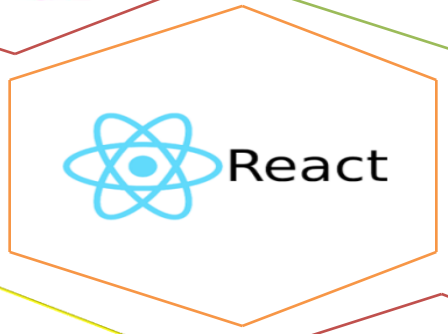
NOW OFFERING!

JAVASCRIPT FRAMEWORKS
CODE DRIVEN CAPACITY BUILDING PROGRAM

Dr. Syed Awase 2016 Session Feedbacks: http://bit.ly/2hhNg58

Reach out to sak@territorialprescience.com/+91.9035433124





Original Series

Dr. Syed Awase also offers Machine learning Stack, R Statistical Stack, .NET Stack, Java Stack, RaspberryPi Stack. Get the pulse of performance from here http://bit.ly/2hhNg58

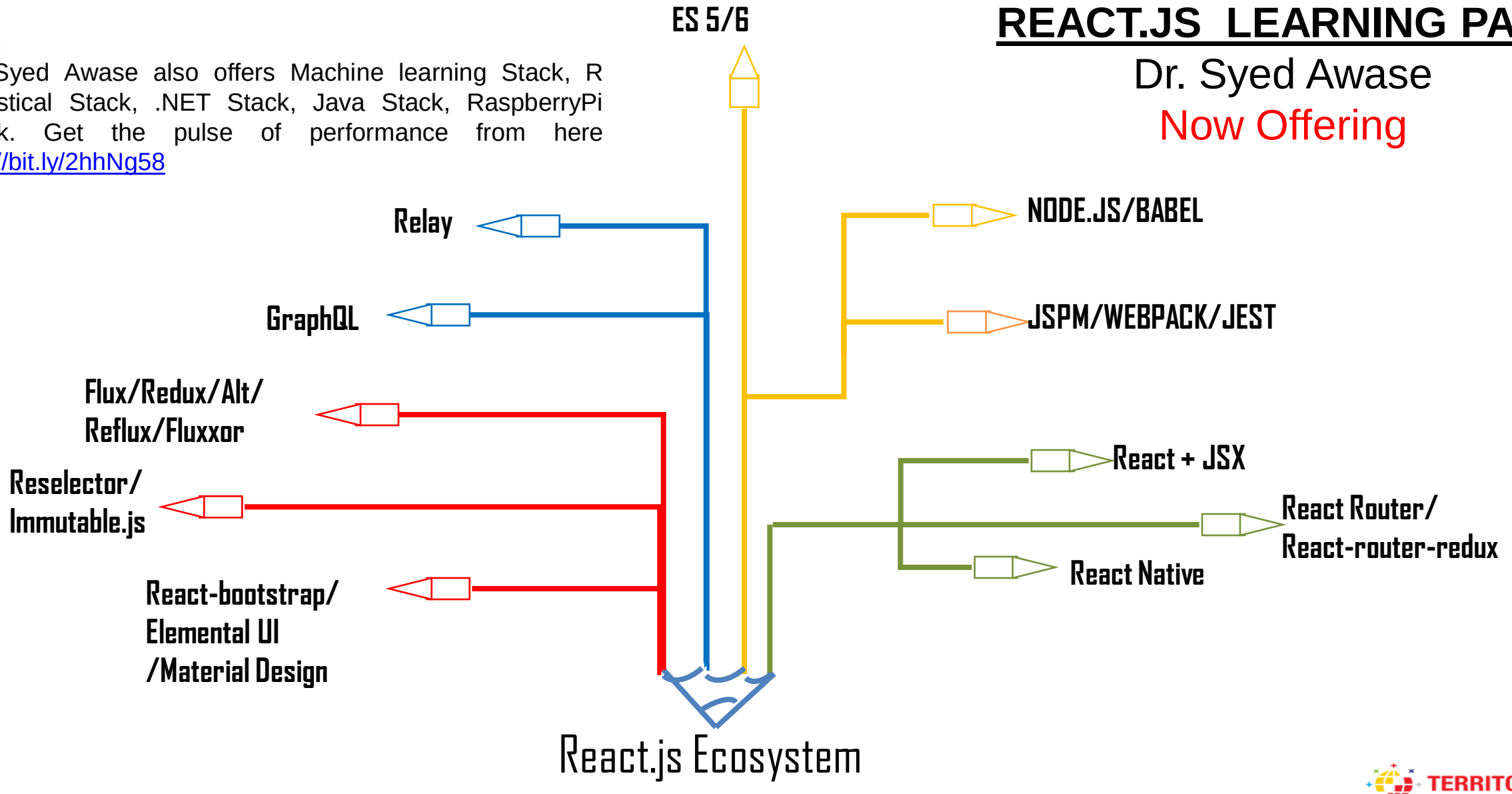
TERRITORIAL PRESCIENCE
provoke thought, invoke action

REACT.JS LEARNING PATH

Dr. Syed Awase

Now Offering

Dr. Syed Awase also offers Machine learning Stack, R Statistical Stack, .NET Stack, Java Stack, RaspberryPi Stack. Get the pulse of performance from here http://bit.ly/2hhNg58

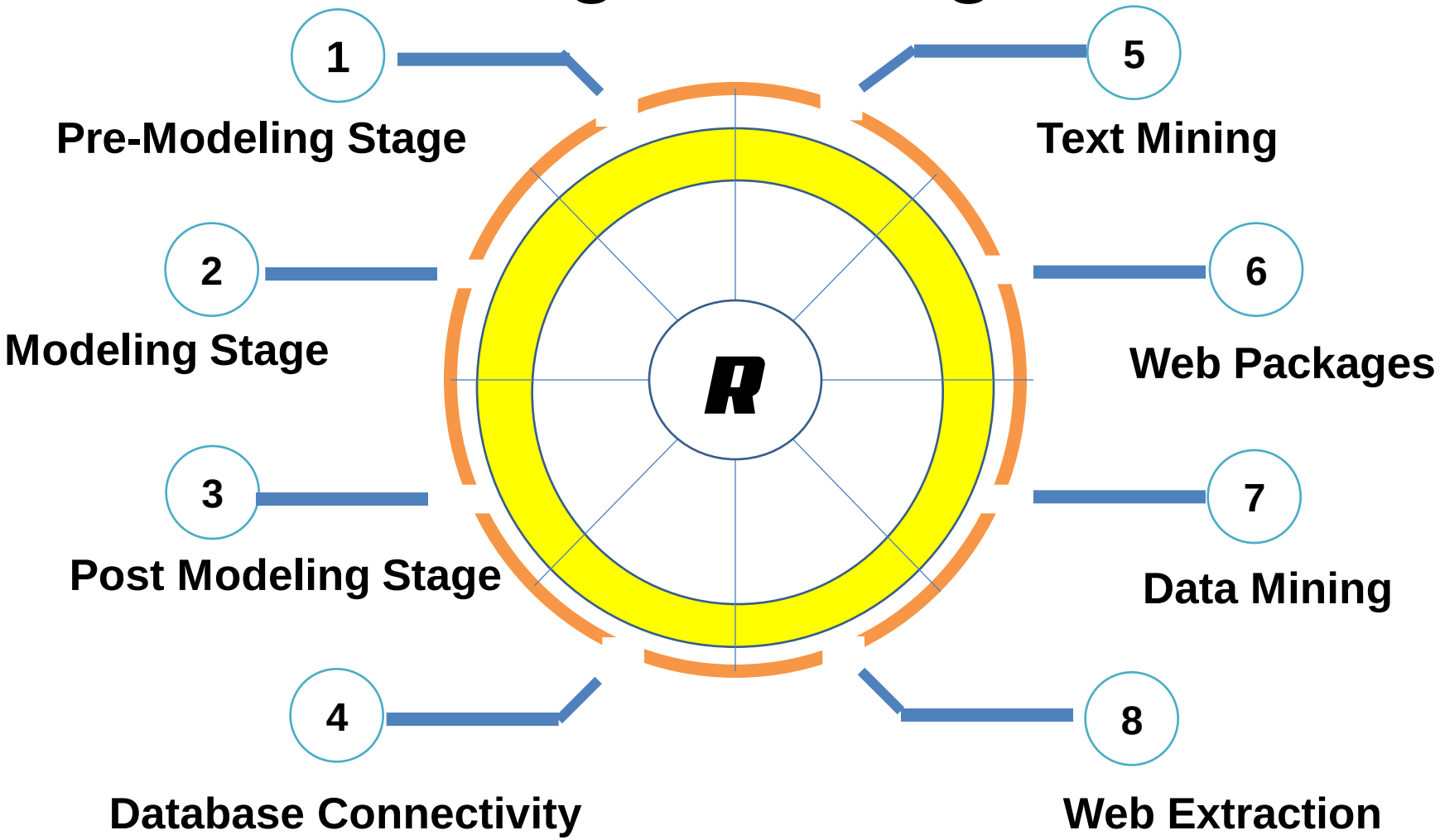


Original Series

R-Statistical Programming

Dr. Syed Awase 2016 Session Feedbacks: http://bit.ly/2hhNg58

Reach out to sak@territorialprescience.com/ +91.9035433124



Original Series

Dr. Syed Awase also offers Machine learning Stack, R Statistical Stack, .NET Stack, Java Stack, RaspberryPi Stack. Get the pulse of performance from here http://bit.ly/2hhNg58

Thank You

We also provide Code Driven Open House Trainings : sak@territorialprescience.com or sak@sycliq.com

Java Technologies

- Core Java
- Hibernate
- Spring Framework
- Play Framework
- Hadoop
- Groovy & Grails

Microsoft Technologies

- C# Core
- Entity Framework
- MVC 5/6
- Web Api
- OWIN/KATANA
- WCF
- WPF

Python

- Python
- Django
- Flask
- Numpy
- Scipy
- Machine Learning

Data Science

- R Statistical Programming
- Julia

SQL and NoSQL

- Oracle
- PostgreSQL
- MSSQL
- MongoDB
- Neo4j
- Redis
- Firebase
- Apache Cassandra

Client-Side Frameworks

- Angular JS 1.5.x
- Angular 2.4.x
- React JS
- KnockOut JS
- VueJS
- Backbone JS
- EMBER JS
- Hapi JS
- METEORJS
- MEANJS
- Coffeescript
- Dart

Others

- LISP
- CLOJURE
- RUST
- GO
- RaspberryPI
- Coming Soon
  - PHP
  - Robotic OS