



F.I.S

提升**产品质量**与**开发效率**的前端解决方案

Quality Software

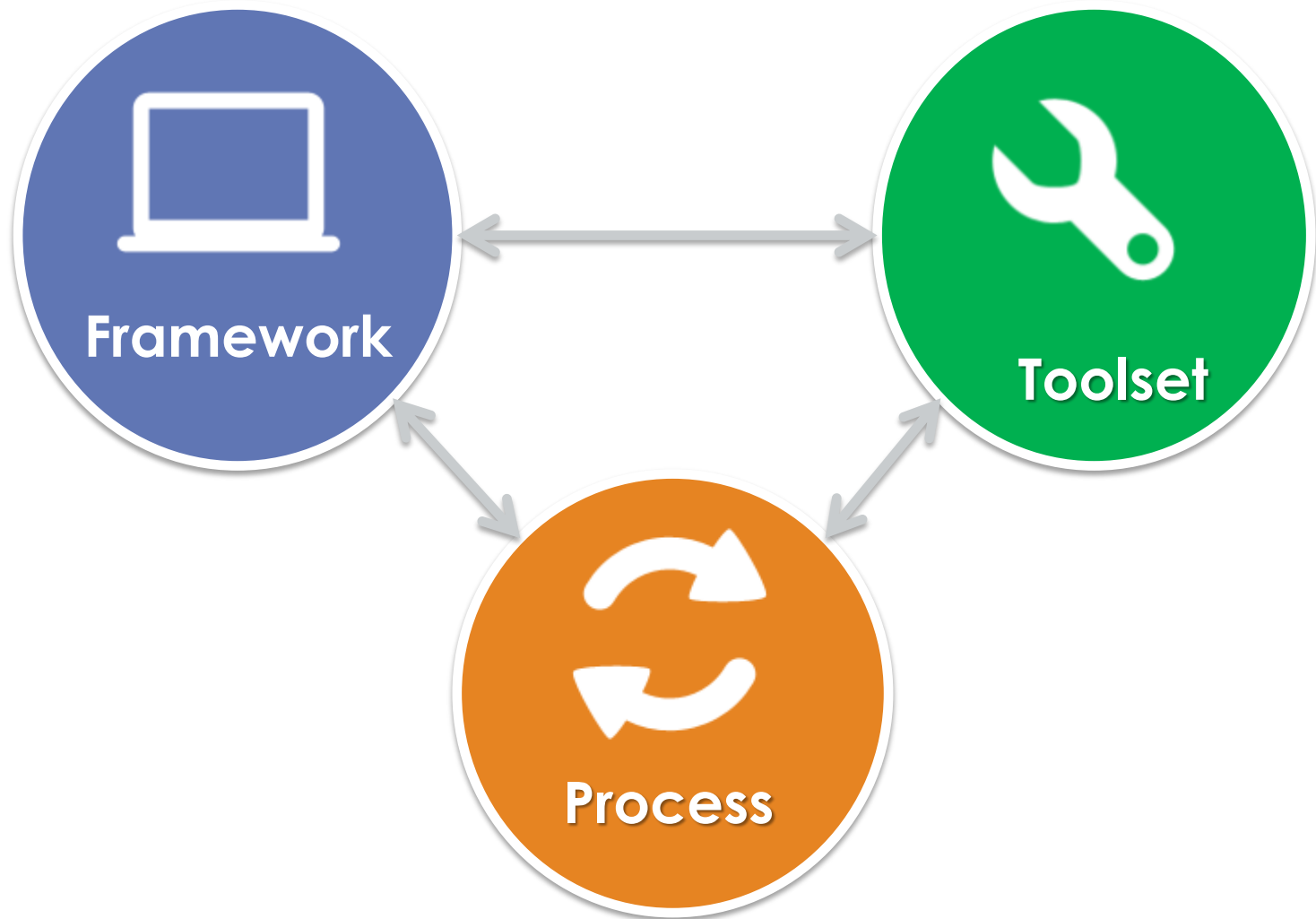
...

- 高可靠性
- 高性能
- 高可维护性
- ...

Quality Takes Time



Front-end Integrated Solution





FIS Toolset

静态资源自动管理

资源定位

资源内嵌

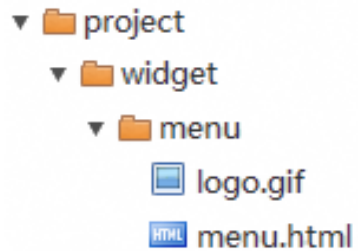
依赖管理

资源合并

资源优化

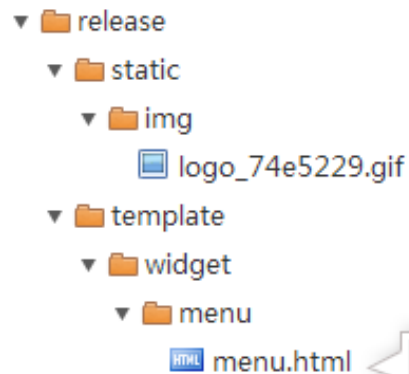
资源部署

- develop:



6 ``

- release:



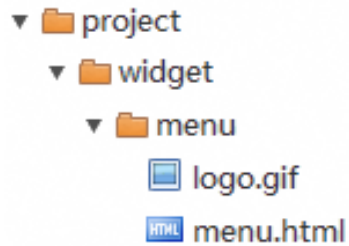
资源定位

//配置文件

```
fis.config.merge({
  roadmap {
    path : [
      ...
      {
        reg : 'widget/**/*.gif',
        release : '/static/img/$&'
      },
      ...
    ]
  }
});
```

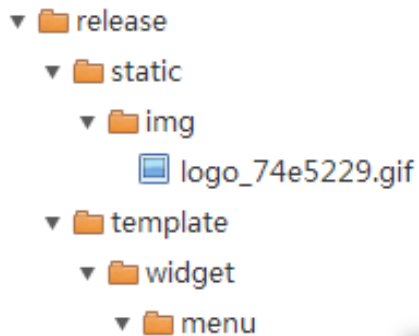
6 ``

- develop :



6 ``

- release :



资源嵌入

6 ``

- 在html中声明依赖

```
<!--
  @require demo.js
  @require "demo.css"
-->
```

- 在javascript中声明依赖

```
//demo.js
/**
 * @require demo.css
 * @require list.js
 */
var $ = require('jquery');
```

- 在css中声明依赖

```
/**
 * demo.css
 * @require reset.css
 */
```

✓ map.json

```
{
  "res" : {
    "demo.css" : {
      "uri" : "/static/css/demo_7defa41.css",
      "type" : "css",
      "deps" : [ "reset.css" ]
    },
    "demo.js" : {
      "uri" : "/static/js/demo_33c5143.js",
      "type" : "js",
      "deps" : [ "demo.css" , "list.js" , "jquery" ]
    },
    "index.html" : {
      "uri" : "/index.html",
      "type" : "html",
      "deps" : [ "demo.js", "demo.css" ]
    }
  },
  "pkg" : {}
}
```

```
//file : fis-conf.js
//开启autoCombine可以将零散资源进行自动打包
fis.config.set('settings.postpackager.simple.autoCombine', true);
```



normalize_5f79cb1.css

/lib



base_920187e.css

/lib



main_217d1b6.css

/lib



todos_133d86a.js

/modules/collections



lib_dbe545c.js

/pkg



app_1b379e7.js

/modules/views



todo_99474cb.js

/modules/models



auto_combine_1_332371e.css

/pkg



lib_dbe545c.js


/pkg





auto_combine_0_02b8ea5.js

/pkg


```
//为所有样式资源开启csssprites  
fis.config.set('roadmap.path', [{  
  reg: '**.css',  
  useSprite: true  
}]);  
//设置csssprites的合并间距  
fis.config.set('settings.spriter.csssprites.margin', 20);
```

 icon_02_3e232bd.gif?__sprite
/lib/icons


 icon_05_5b35a82.png?__sprite
/lib/icons

 icon_04_fccba9e.png?__sprite
/lib/icons

 icon_03_85d0109.gif?__sprite
/lib/icons

 icon_01_766089e.gif?__sprite
/lib/icons



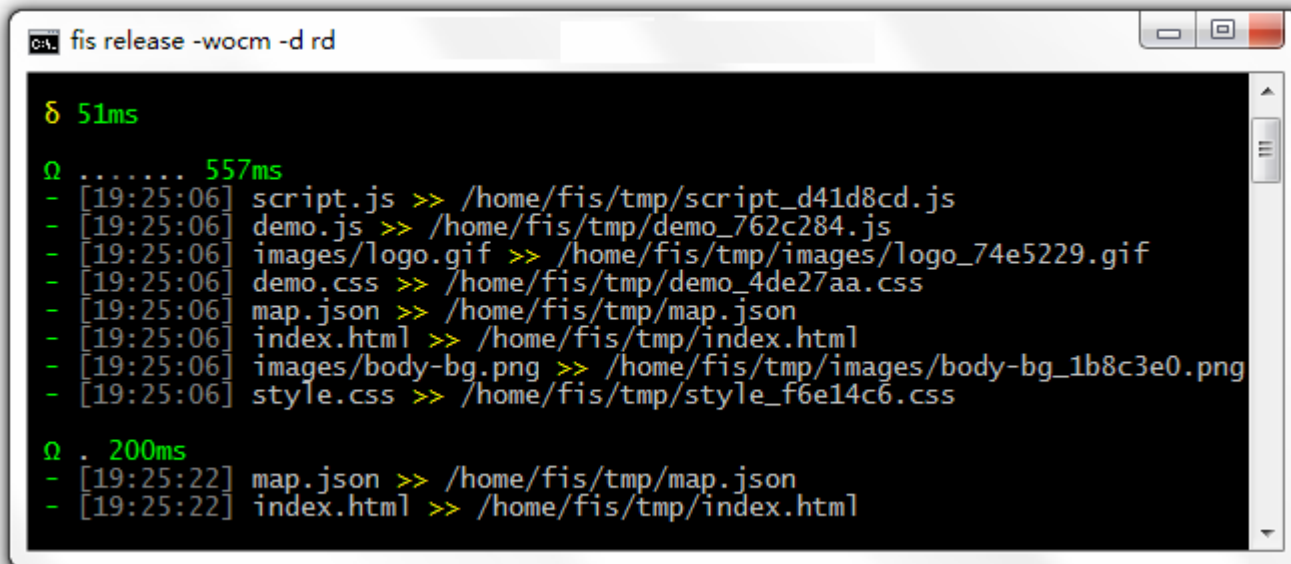
 base_z_b5bdb52.png
/lib



fis release [options]

文件监听 + 优化 + 时间戳 + CDN + 测试 + 校验 + 合并 + 自动部署

fis release -womDtlp -d rd,qa



```

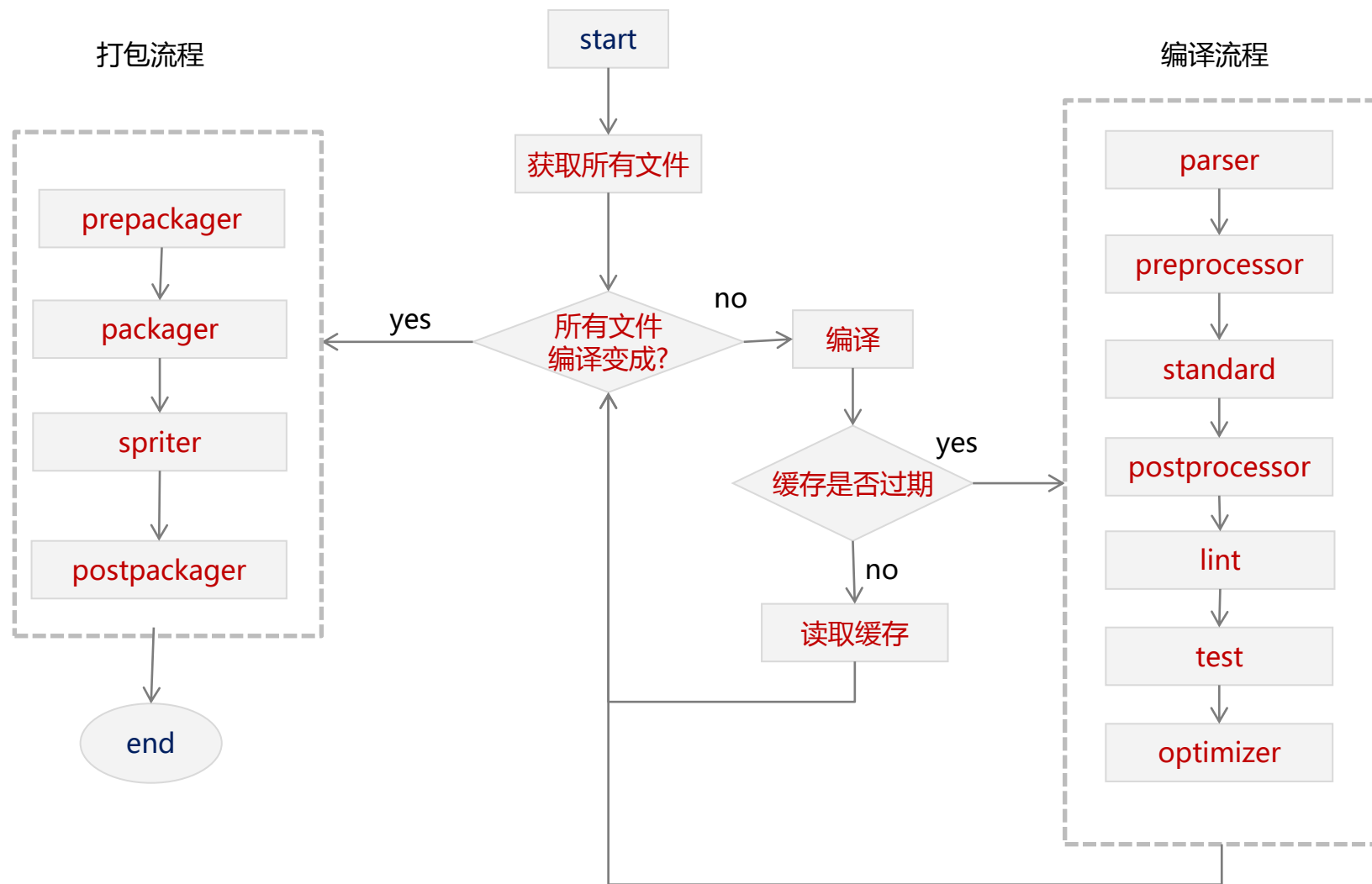
fis release -wom -d rd

δ 51ms
Ω ..... 557ms
- [19:25:06] script.js >> /home/fis/tmp/script_d41d8cd.js
- [19:25:06] demo.js >> /home/fis/tmp/demo_762c284.js
- [19:25:06] images/logo.gif >> /home/fis/tmp/images/logo_74e5229.gif
- [19:25:06] demo.css >> /home/fis/tmp/demo_4de27aa.css
- [19:25:06] map.json >> /home/fis/tmp/map.json
- [19:25:06] index.html >> /home/fis/tmp/index.html
- [19:25:06] images/body-bg.png >> /home/fis/tmp/images/body-bg_1b8c3e0.png
- [19:25:06] style.css >> /home/fis/tmp/style_f6e14c6.css

Ω . 200ms
- [19:25:22] map.json >> /home/fis/tmp/map.json
- [19:25:22] index.html >> /home/fis/tmp/index.html
```

- 有效的分离开发路径与部署路径之间的关系
 - 工程师不再关心资源部署到线上之后去了哪里，变成了什么名字，这些都可以**通过配置来指定**
- 代码具有很强的可移植性
 - 由于开发路径与部署路径对工程师透明，因此组件的资源依赖全部都是**相对路径定位**的，这样，对于两个同样使用 fis 作为开发平台的团队，即便他们的部署方式完全不同也可以有效移植代码。
- 轻松实现md5、域名添加等功能
 - 工程师完全不用关心上线后资源的静态服务器域名是什么
 - 资源会全部自动添加md5作为版本戳，服务器可以开启**强缓存**、回滚时不需要回滚静态资源，只须回滚html或模板即可

管道式处理流程



使用 FIS ，轻松定制属于你的解决方案

FIS Plus

Yogurt

Jello

pure

gois



模板框架解决方案，内置静态资源运行时管理和加载框架，提高服务端的渲染效率和并行度，使得首屏及核心功能最快展现，适用于网络高延迟/低带宽、国际化等多种业务场景。

[查看详细](#)

FIS Framework

Plus



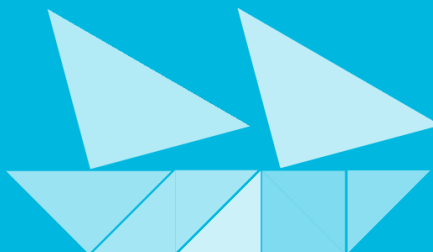
基于 **PHP/Smarty** 实现的 fis 展现层解决方案

Pure



纯前端 fis 展现层解决方案，不依赖于后端

Yogurt



基于 **NodeJS/Express.js** 实现的 fis 前后端一体化解决方案

Jello



基于 **Java/VelocityEngine** 实现的 fis 展现层解决方案

Gois



基于 **Go/Martini** 实现的 fis 展现层解决方案

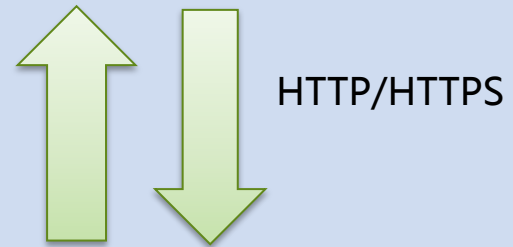


NodeJS 前后端一体化框架(Yogurt)

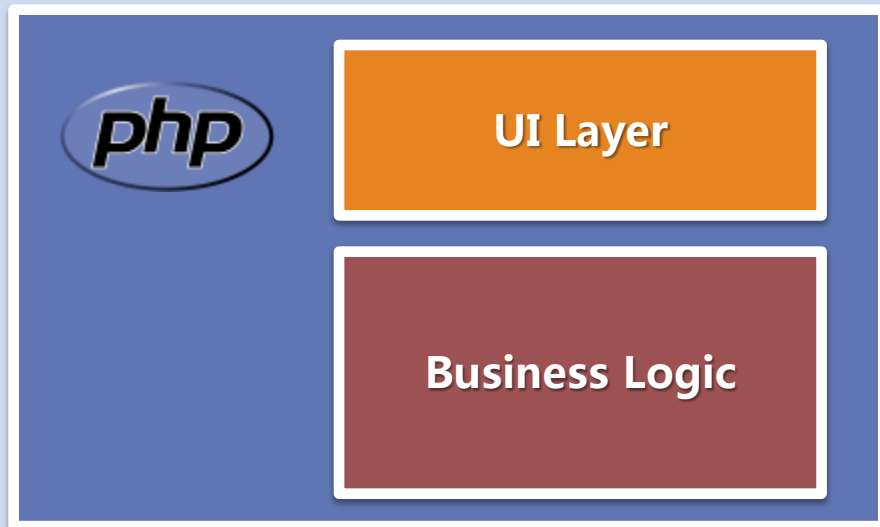
Front-End



Browser

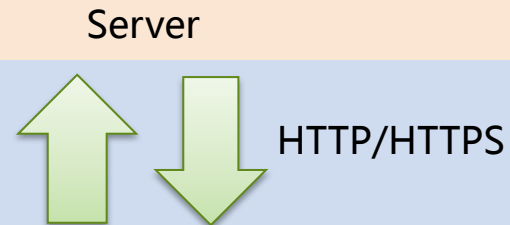
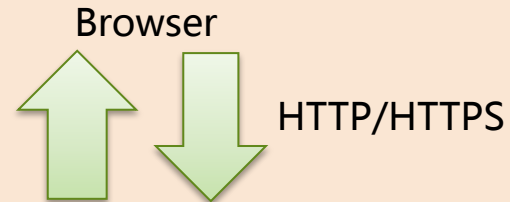


Back-End



Server

Front-End

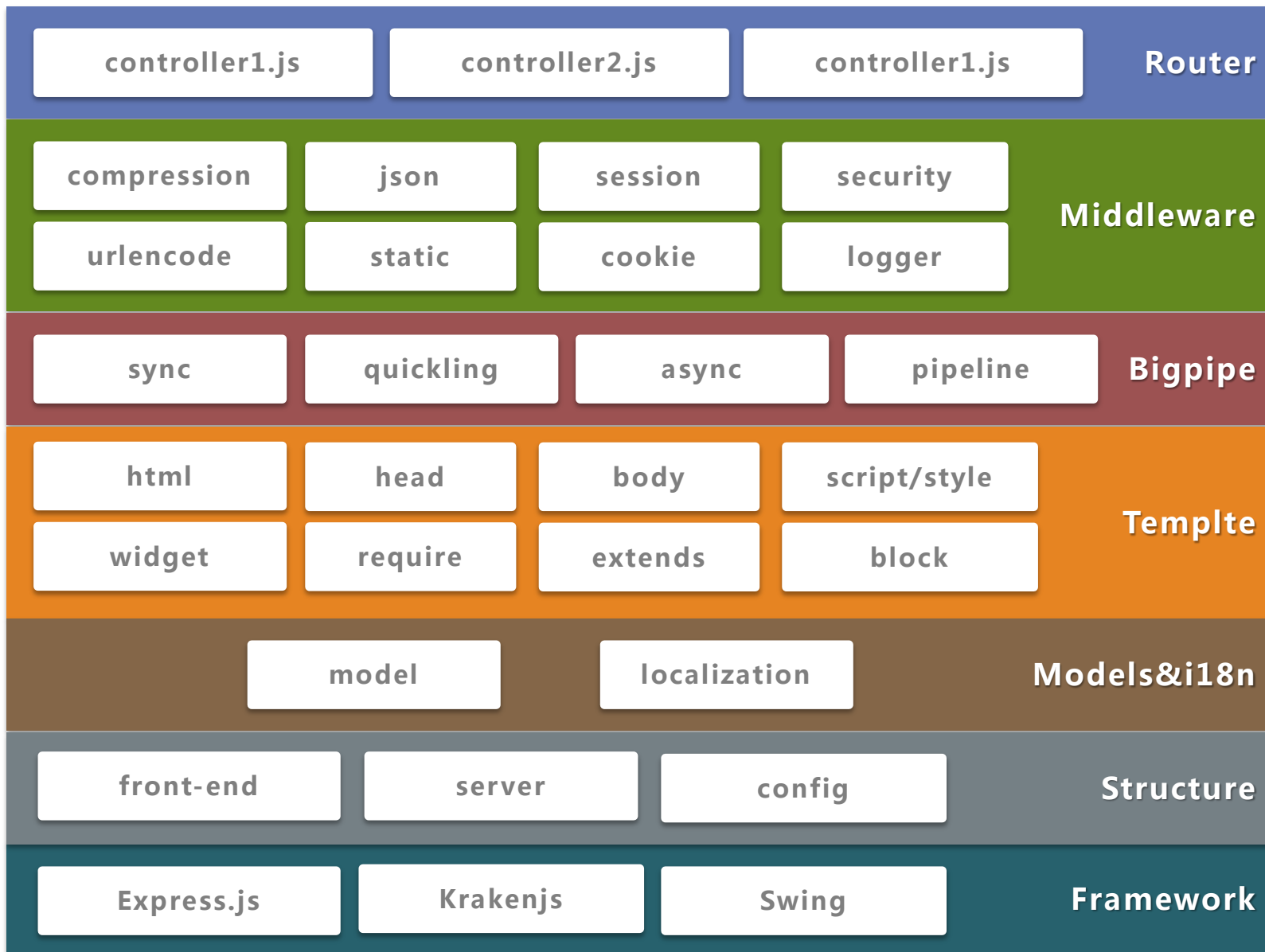


Back-End



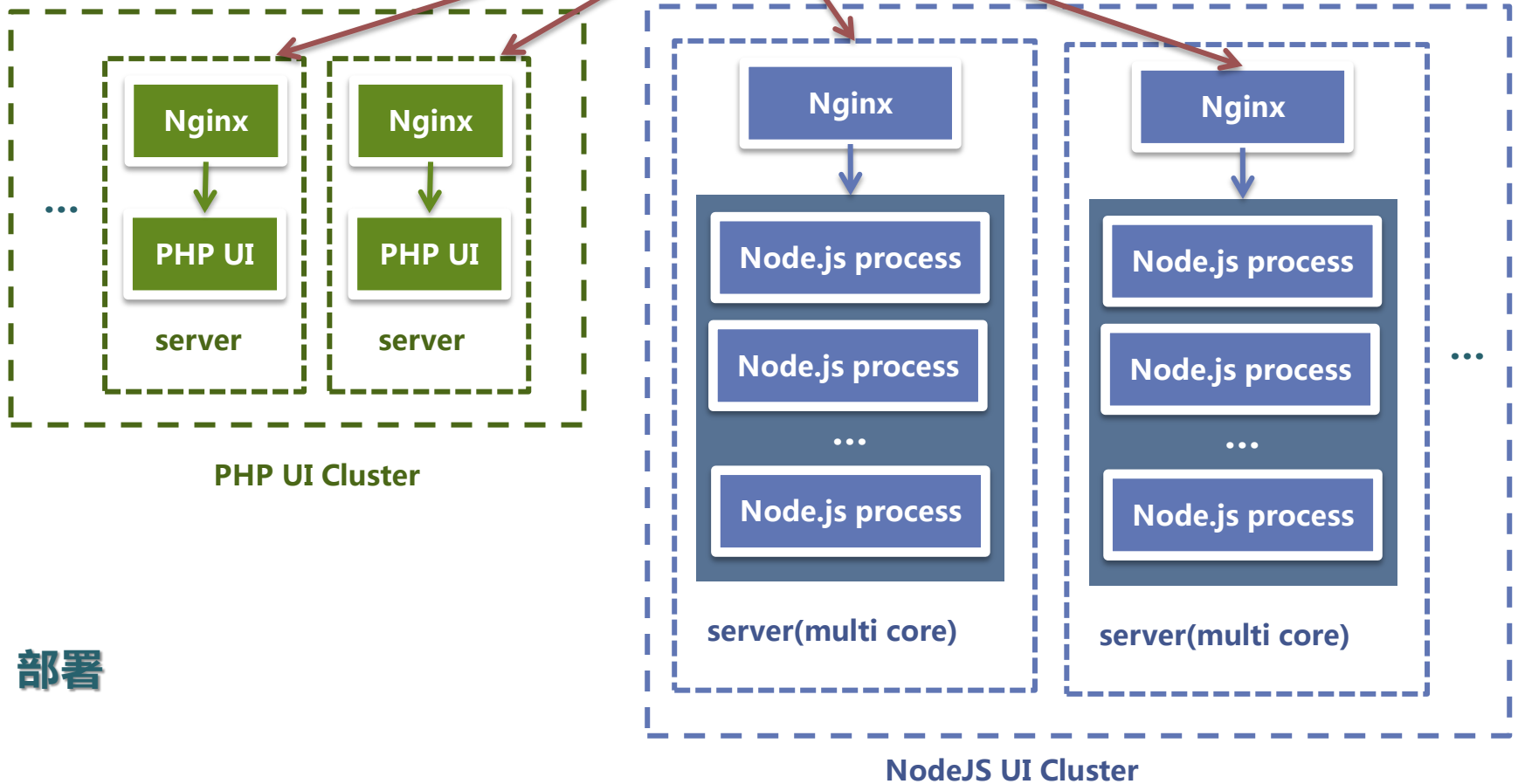
Server

开发



Request

Load Balancer(Transmit)



monitor

startup/reload

memwatch

error handling

forewarning

daemon

log

load balancer

heartbeat check

0s downtime reload

运维 Yog-pm (extends pm2)

Solar

FIS 云服务平台



静态资源自动合并系统

根据线上资源使用情况，从性能优化角度出发自动合并资源，解决人力成本，提升产品性能

[查看详情](#)

Feature Flag系统

无需分支上线，轻松控制线上新功能发布时机、面向人群、流量等。

[查看详情](#)

Lights前端资源聚合平台

便捷、易用的资源安装、发布、搜索，管理工具。快速共享团队资源，提升开发效率。

[查看详情](#)

FIS编译器插件管理平台

轻松管理多台线上编译机的FIS插件的同步、安装、更新等，让编译器不再黑盒，一目了然。

[查看详情](#)

FIS Process



↓ 30% 静态资源大小
↑ 10% 访问性能

FIS静态资源自动合并服务(Auto-Pack)


```
<html>
  <link href="A.css">
  <link href="B.css">
  <link href="C.css">
  <div>html of A</div>
  <div>html of B</div>
  <div>html of C</div>
</html>
```



```
<html>
  <link href="A-B-C.css">
  <div>html of A</div>
  <div>html of B</div>
  <div>html of C</div>
</html>
```



```
<html>
  <link href="A-B-C.css">
  <div>html of A</div>
  <div>html of B</div>
  {if $user_has_C}
    <div>html of C</div>
  {/if}
</html>
```

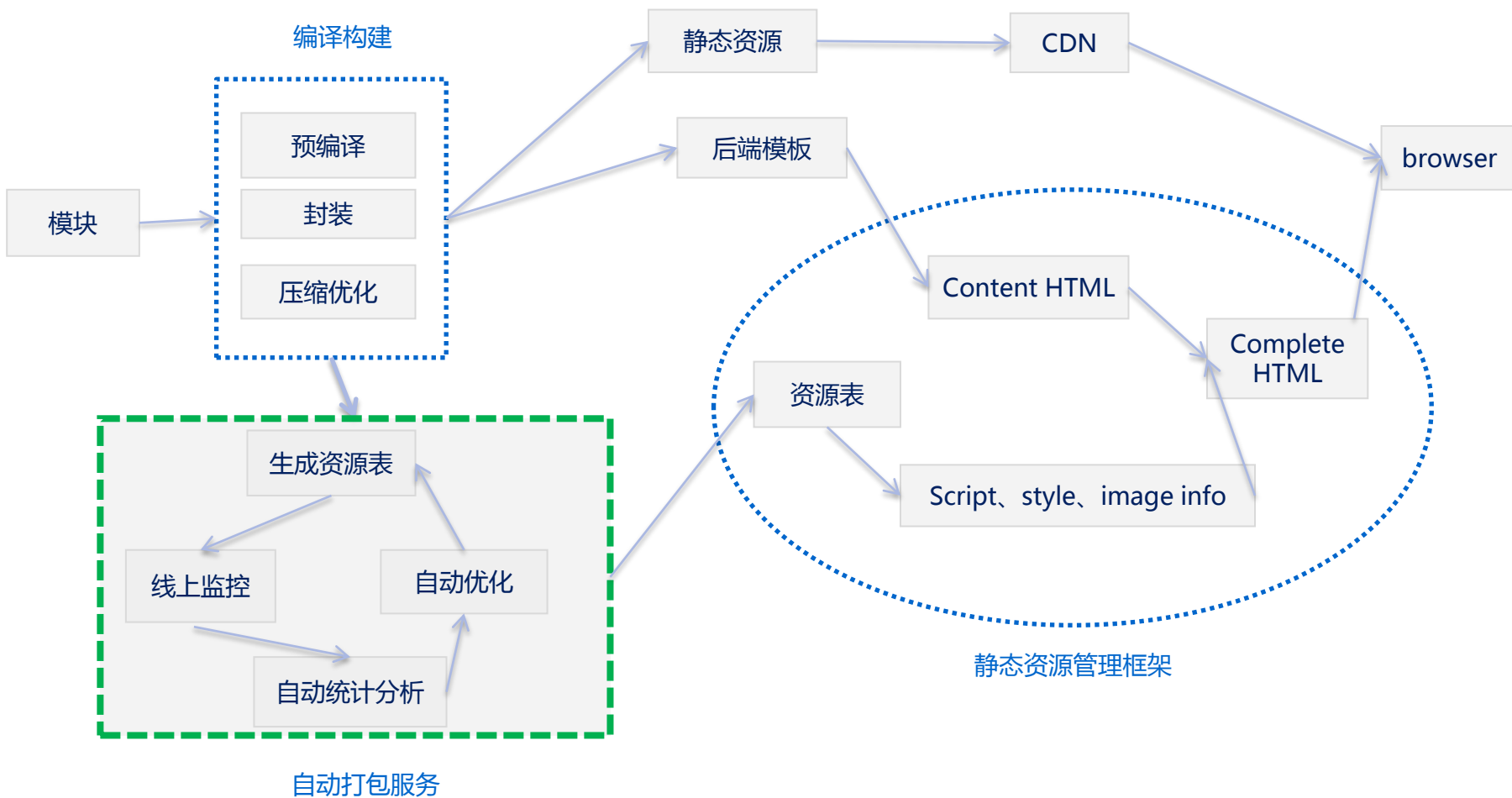


```
<html>
  <link href="A-B-C-D-E-F-G-H....css">
  <div>html of A</div>
  <div>html of B</div>
  ...
  {if $not_used_F}
    <div>html of E</div>
  {else}
    <div>html of F</div>
    <div>html of G</div>
  {/if}
  ...
</html>
```



```
<html>
  <link href="A-B-C.css">
  <div>html of A</div>
  <div>html of B</div>
  {*if $user_has_C}
    <div>html of C</div>
  {/if*}
</html>
```





资源合并算法

- **合并收益** : 对于同时使用A和B的页面节省了网络来回时间(RTT)
- **合并损失** : 对于只使用A的页面, 浪费的B的大小(损失的大小 / 下载速度)
- **纯收益** : 合并收益 - 合并损失

	Page_1	Page_2	Page_3	Page_4	Page_5
访问量	10M	1M	200K	10K	1K
A.Js (1KB)	√	√	√	√	√
B.Js (1KB)	√	√	√	√	√
C.Js (300B)	√	√			
D.Js (2KB)					√
E.Js (700B)		√	√		
F.Js (600B)		√	√	√	√

区分首屏和延迟加载

区分网络

区分国家

...

Auto-Pack

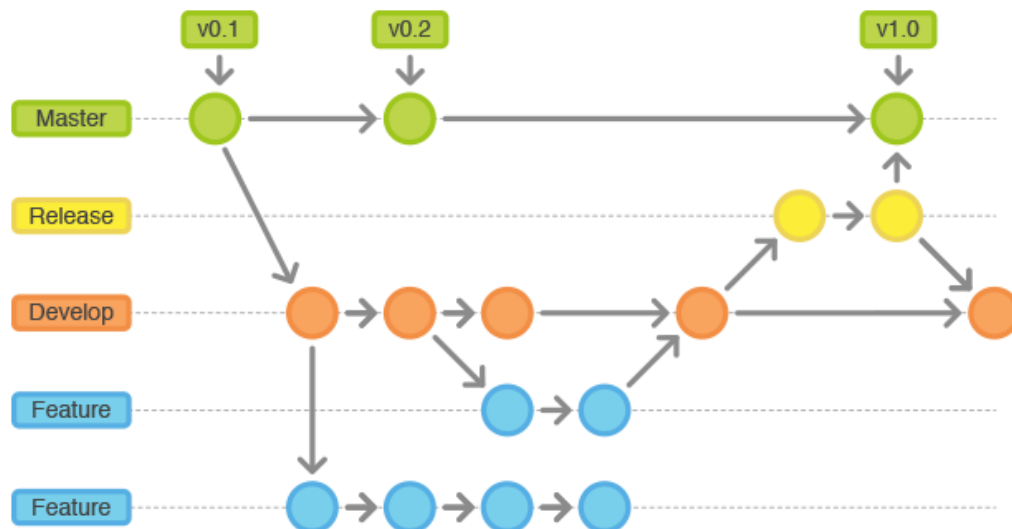
- 静态资源自适应优化合并服务
 - 根据网站页面pv以及页面静态资源使用情况，自动计算静态资源合并方案，减少人工管理静态资源成本和风险
 - 从网络请求、首屏渲染等方面优化网站性能、减少服务器开销
 - 对工程师完全透明，解决手工维护的未及时排除废弃资源、不可持续、成本大等问题



功能发布控制系统(Feature-Flag)

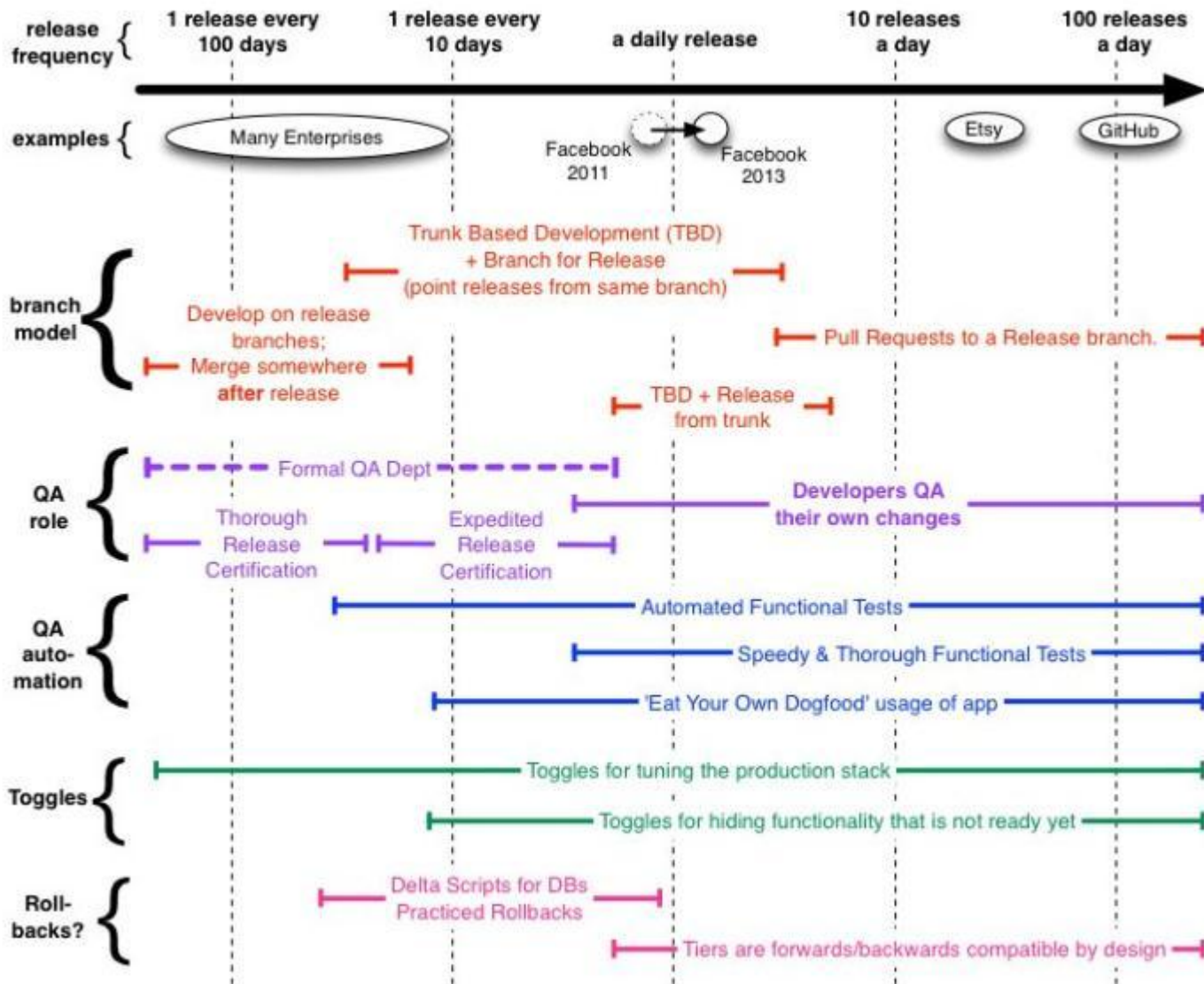
精准发布，控制自如

Feature Branches

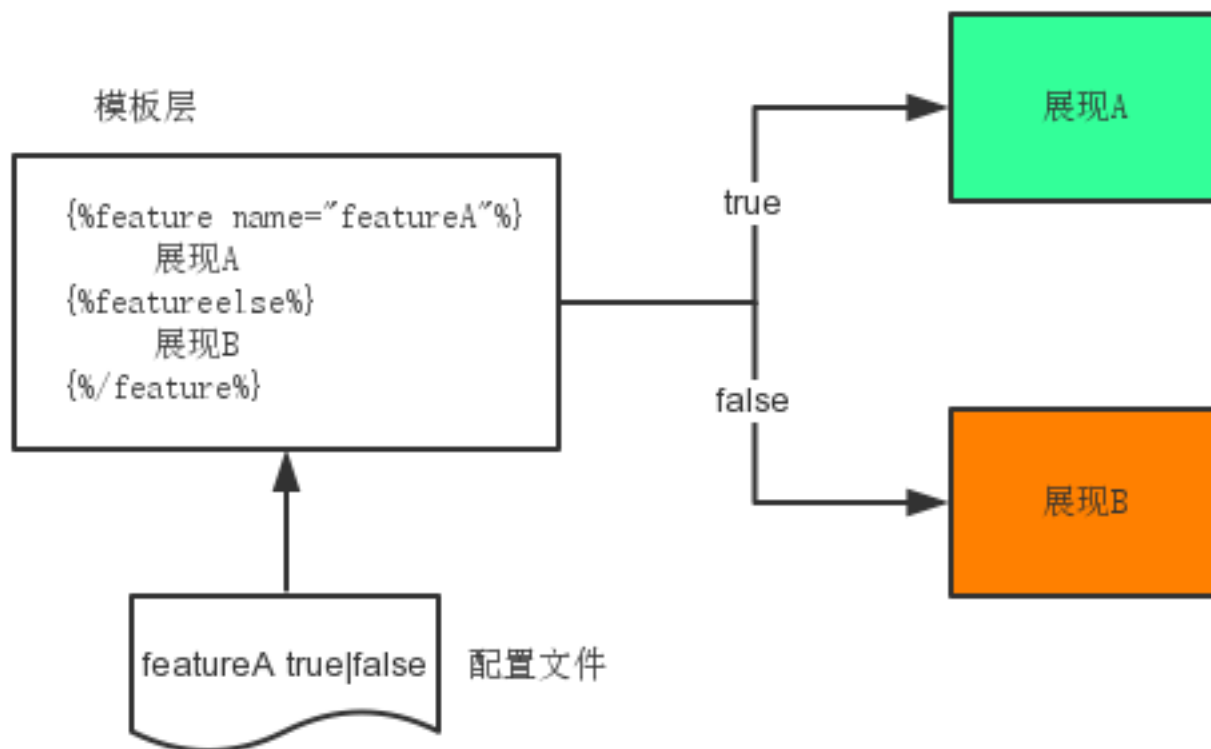


- 分支分出去时间越长往往代码merge难度越大、风险和成本越高
- 在一个分支中修改了函数名字可能会引入大量编译错误，重构成本很高
- 一旦代码库中 exist 了分支，无法很好的支持持续集成，迭代速度受影响
- 有多个 feature branches 的时候，无法测试这多个 feature 之间的影响

Facebook's Trunk Based Development



Feature-Flag



根据各种**场景**和**条件**配置控制是否展现页面某一区域或功能，不用重新发布代码

Feature-Flag

- Feature-Flag框架
 - 快速回滚
 - 小流量
 - A/B测试
 - 特定时间发布
 - 特定区域发布
 - 主干开发
- Flag管理平台, 可视化管理产品中的所有 feature flag
- 小流量评估平台, 结合 feature flag 自动分析、评估小流量的效果和收益

Feature-Flag 使用注意

- 如果某个功能最终不上线，后续需要手工删除相关代码
- 会出现因为配置错误某个 feature 没有完成就出现在线上
- 需要控制 flag 数量，有可能会被滥用

开源

- 百度FEX-FIS团队
- 用户：百度、阿里、腾讯、UC、小米、去哪...
- QQ交流群：315973236
- web site： <http://fis.baidu.com/>
- github： <https://github.com/fex-team/fis>
- Blog： <http://fex.baidu.com/>
- 招聘： <http://fex.baidu.com/we-need-you/>

<Thank You!>

@walterShen