Vue 驱动的
小程序 + H5 同构开发框架

张文韬

VueConf 上海 2019.06

About me

张文韬

前端工程师 @百度-前端技术部

搜索垂类产品横向技术团队

负责 Vue 驱动的多端开发框架 Mars

Github @allen-zh

Why [another] wheel?

我们希望这样的框架……

```
wheels.some(wheel => wheel.hasAllFeatures([
```

支持 H5 同构并提供完整适配开发方案,

支持小程序自定义组件,

支持尽量多的 Vue 特性,

极致性能优化

```
]) // => false
```

Why ~~[another] wheel~~?

我们希望这样的框架……

```
wheels.filter(wheel => wheel.hasAllFeatures([
```

支持 H5 同构并提供完整适配开发方案,

支持小程序自定义组件,

支持尽量多的 Vue 特性,

极致性能优化

```
]) // => ['Mars']
```

Mars

Vue 驱动

一处开发，多端运行

已支持 百度智能小程序、微信小程序、H5 (PWA)



Github https://github.com/max-team/Mars

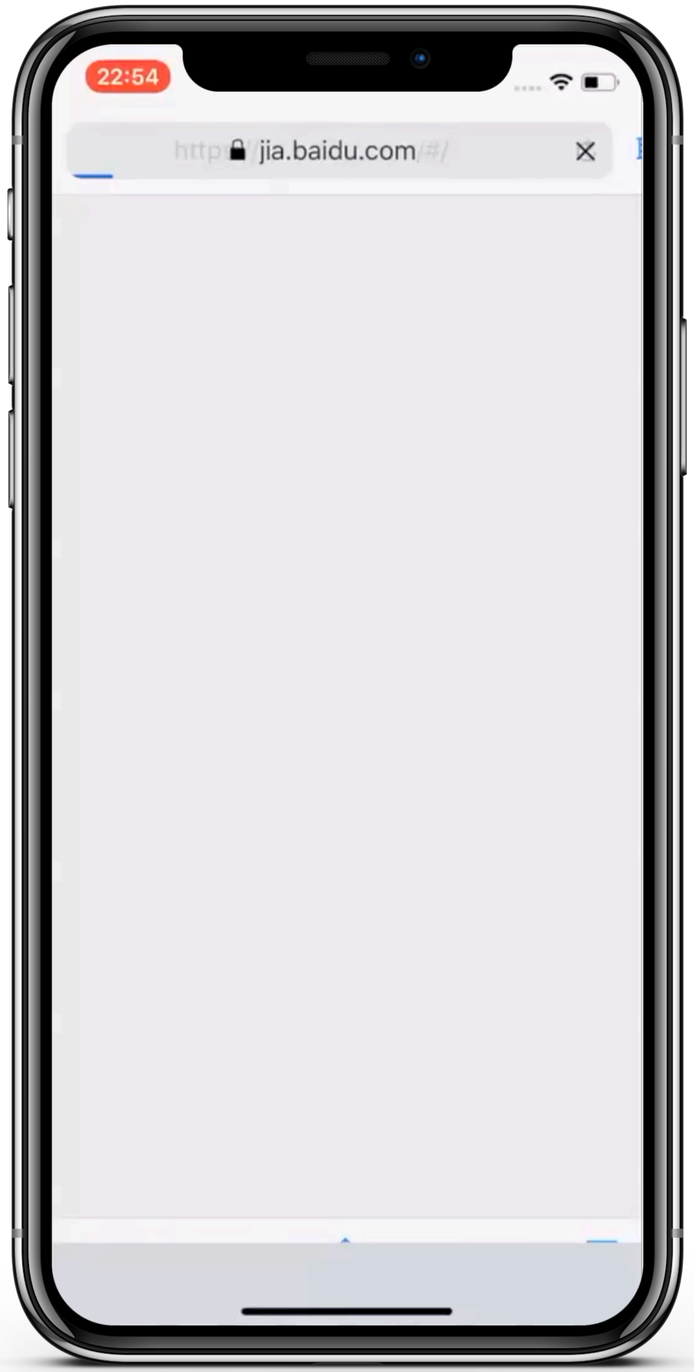
文档 https://max-team.github.io/Mars/

Mars

H5 同构效果

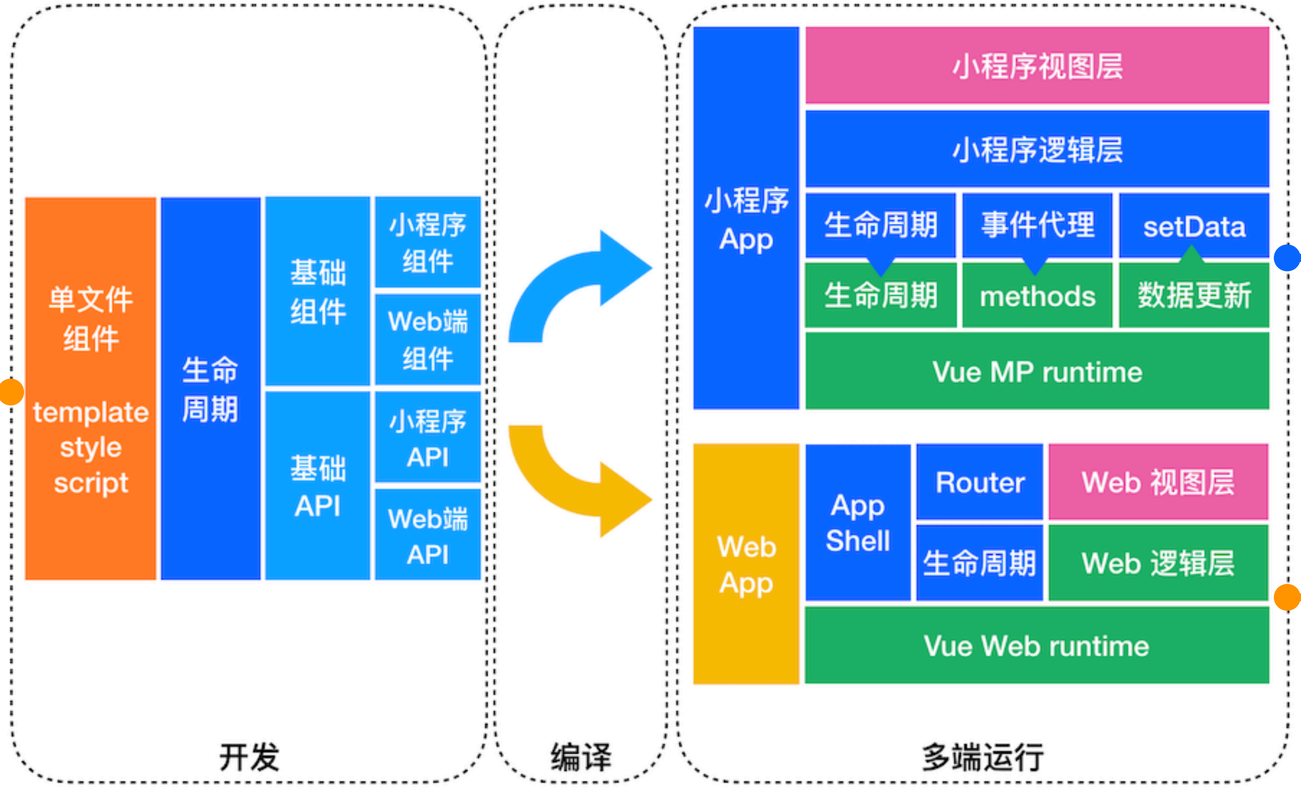
SHOWCASE：百度家装【装馨家】

百度智能小程序	H5



How

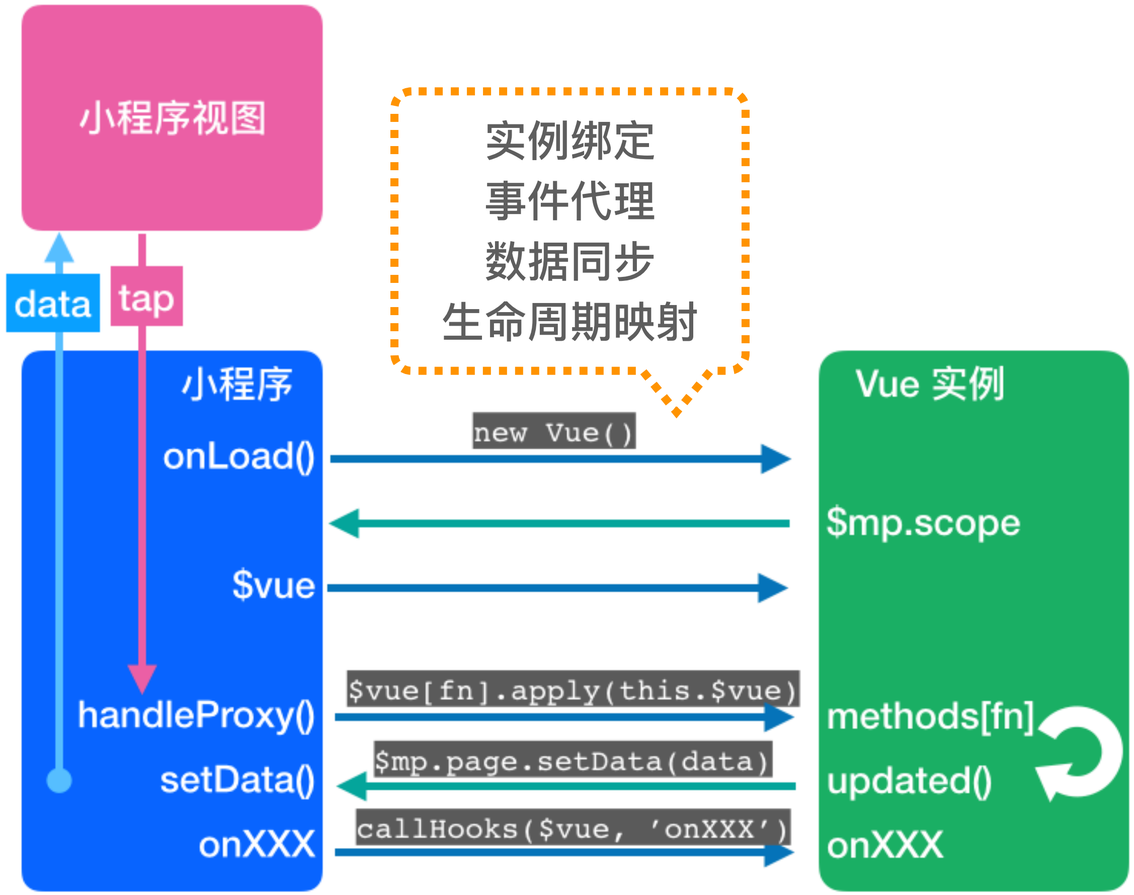
Vue 单文件组件规范
标准生命周期规范
基础组件、API 规范



小程序运行时适配
实现 Vue 驱动

同构 App 框架、路由
同构组件、API、钩子函数
标准 Vue 项目
体验优化 (PWA等支持)

Vue drive


小程序视图
data
tap
小程序
onLoad()
$vue
handleProxy()
setData()
onXXX
实例绑定
事件代理
数据同步
生命周期映射
new Vue()
$vue[fn].apply(this.$vue)
$mp.page.setData(data)
callHooks($vue, 'onXXX')
Vue 实例
$mp.scope
methods[fn]
updated()
onXXX

Vue drive with …

core/platform

轻量级定制

修改总行数 < 50 行

modules

通过运行时 module 结合

VNode 在小程序支持过滤器

compiler

通过 modules 编译接口

扩展 render 函数

directives

通过 v-animation 指令在

H5 实现 createAnimation

Vue drive core

2 Lines

修改 inBrowser 判断

保留 patch

15 src/core/util/env.js

```
 export const hasProto = '__proto__' in {}

 // Browser environment sniffing
+// export const inBrowser = typeof window !== 'undefined'
+export const inBrowser = false
 export const inWeex = typeof WXEnvironment !== 'undefined' &&
```

4 src/platforms/web/runtime/index.js

```
 /* @flow */
+/* eslint-disable no-unused-vars */

 import Vue from 'core/index'
 import config from 'core/config'
 extend(Vue.options.components, platformComponents)

 // install platform patch function
+// Vue.prototype.__patch__ = inBrowser ? patch : noop
+Vue.prototype.__patch__ = patch
```

Vue drive platform

30 Lines

移除 DOM 相关模块

修改 node-ops

期待 Vue 3.0

createRenderer() API

8 src/platforms/web/runtime/directives/index.js

```
+// import model from './model'
+// import show from './show'

 export default {
+  // model,
+  // show
 }
```

8 src/platforms/web/runtime/components/index.js

```
+// import Transition from './transition'
+// import TransitionGroup from './transition-group'

 export default {
+  // Transition,
+  // TransitionGroup
 }
```

48 src/platforms/web/runtime/modules/index.js

```
+// import attrs from './attrs'
+// import klass from './class'
+// import events from './events'
+// import domProps from './dom-props'
+// import style from './style'
+// import transition from './transition'
+
+// export default [
+//   attrs,
+//   klass,
+//   events,
+//   domProps,
+//   style,
+//   transition
+// ]
```

50 src/platforms/web/runtime/node-ops.js

```
 /* @flow */
+/* eslint-disable no-unused-vars */

+// import { namespaceMap } from 'web/util/index'

 export function createElement (tagName: string, vnode: VNode): Element {
+  return vnode
+  // const elm = document.createElement(tagName)
+  // if (tagName !== 'select') {
+  //   return elm
+  // }
+  // // false or null will remove the attribute but undefined will not
+  // if (vnode.data && vnode.data.attrs && vnode.data.attrs.multiple !== undefined) {
+  //   elm.setAttribute('multiple', 'multiple')
+  // }
+  // return elm
```

Vue drive 案例: 数据变更增量更新

Vue core

10 Lines

observer 增加更新标记

实现增量更新优化性能

1 src/core/observer/array.js

```
40        if (inserted) ob.observeArray(inserted)
41        // notify change
42        ob.dep.notify()
43  +     ob.__changed__ = true
44        return result
45      })
46    })
```

7 src/core/observer/index.js

```
189         }
190         childOb = !shallow && observe(newVal)
191         dep.notify()
192  +      if (!obj._isVue && obj.__ob__) {
193  +        const ob = obj.__ob__;
194  +        ob.__changedKeys__ = ob.__changedKeys__ ?
ob.__changedKeys__ : {}
195  +        ob.__changedKeys__[key] = true
196  +      }
197       }
198     })
199   }
232     }
233     defineReactive(ob.value, key, val)
234     ob.dep.notify()
235  +  ob.__changedKeys__ = ob.__changedKeys__ ? ob.__changedKeys__
: {}
236  +  ob.__changedKeys__[key] = true
237     return val
```

Vue drive 案例: 数据变更增量更新

运行时

运行时

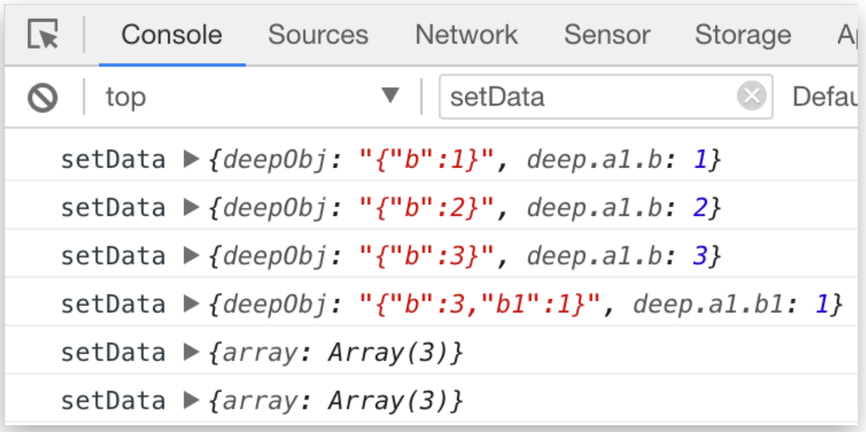
找到更新了的数据

计算 keyPath

增量数据更新

```
function getChangedData(vm, _data, keyPath = '', ret = {}) {
  const { __ob__: ob } = _data
  if (!ob) {
    return ret
  }
  const { __changedKeys__: changedKeys } = ob
  vm.__mpKeyPath = vm.__mpKeyPath || {}

  if (ob.__changed__ || ob.__changedKeys__) {
    vm.__mpKeyPath[ob.dep.id] = ob
  }
  if (ob.__changed__) {
    ret[keyPath] = _data
  } else {
    Object.keys(_data).forEach(key => {
      const data = _data[key]
      const path = (keyPath ? `${keyPath}.` : '') + key
      if (changedKeys && changedKeys[key]) {
        ret[path] = data
      } else if (data instanceof Object) {
        getChangedData(vm, data, path, ret)
      }
    })
  }
  return ret
}
```



Vue drive 案例: 实现过滤器

设计

为小程序过滤器设计的数据结构

```
<view>
    <view :text="reverse(btnText) | capitalize">
        {{btnText | capitalize}}
        <view>{{btnText}}</view>
    </view>
</view>
```

```
<view>
    <view text="{{ _f_[0]._p.text }}">
        {{ _f_[0]._t[0] }}
        <view>{{btnText}}</view>
    </view>
</view>
```

```
// init:
// btnText === 'hello'
{
    _f_: {
        0: {
            _t: ['Hello'],
            _p: {
                text: 'Olleh'
            }
        }
    }
}

// update:
// btnText === 'olleh'
{
    '_f_.0._t.0: 'Olleh',
    '_f_.0._p.text: 'Hello',
    '_f_.1': {
        _t: [],
        _p: {}
    }
}
```

Vue drive 案例: 实现过滤器

compiler

```
<view>
    <view :text="reverse(btnText) | capitalize">
        {{btnText | capitalize}}
        <view>{{btnText}}</view>
    </view>
</view>
```

```
<view>
    <view text="{{ _f_[0]._p.text }}">
        {{ _f_[0]._t[0] }}
        <view>{{btnText}}</view>
    </view>
</view>
```

```
return _c(
  'view',
  [
    _c('view', {
      f: {
        fid: 0,
        t: ["\n        " + _vm._s(_vm._f("capitalize")(_vm.btnText)) + "\n
        p: ["text"]
      },
      attrs: {
        "text": _vm._f("capitalize")(_vm.reverse(_vm.btnText))
      }
    },
    [
      _vm._v("\n        " + _vm._s(_vm._f("capitalize")(_vm.btnText)) + "\n
      _c('view', [_vm._v(_vm._s(_vm.btnText))])
    ])
  ]
)
```

Vue drive 案例: 实现过滤器

运行时

从 Vue 实例上获取带有过滤器的 VNode 节点以及结构化数据

```
function getFiltersData(vm, $mp, data = {}) {
    if (vm._fData) {
        Object.keys(vm._fData).forEach(k => {
            // ...
            const curData = originFData && originFData[k];
            if (!curData) {
                if (props) {
                    kData._p = kData._p || {};
                    props.forEach(key => {
                        kData._p[key] = propsData[key];
                    });
                }
                if (texts) {
                    kData._t = kData._t || {};
                    texts.forEach((t, index) => {
                        kData._t[index] = t + '';
                    });
                }

            }
            else {
                // compare texts
                if (texts) {
                    texts.forEach((t, index) => {
                        compareAndSetData(k, t + '', curData._t[index], `_t.${index}`, data);
                    });
                }
                // compare props
```

```
// init:
// btnText === 'hello'
{
    _f_: {
        0: {
            _t: ['Hello'],
            _p: {
                text: 'Olleh'
            }
        }
    }
}

// update:
// btnText === 'olleh'
{
    '_f_.0._t.0: 'Olleh',
    '_f_.0._p.text: 'Hello',
    '_f_.1': {
        _t: [],
        _p: {}
    }
}
```

Vue drive 案例: 实现过滤器

运行时

Vue runtime modules

通过 **运行时 modules** 将 VNode 上的过滤器值与实例关联，方便从实例上获取过滤器值

42 ■■■■■ src/core/vdom/modules/filters.js

```
+/**
+ * @file filters.js
+ * @description bind vnode to vm for Mars filters
+ * @author zhangwentao <winty2013@gmail.com>
+ */
+
+function bindFilter(vnode) {
+    let {data, context} = vnode;
+    if (data && data.f && data.f.fid !== undefined) {
+        context._fData = context._fData || {};
+        context._fData[data.f.fid] = vnode;
+    }
+}
+
+function unbindFilter(vnode) {
+    let {data, context} = vnode;
+    if (data && data.f && data.f.fid !== undefined) {
+        if (context._fData && context._fData[data.f.fid]) {
+            context._fData[data.f.fid] = null;
+        }
+    }
+}
```

```
+export default {
+    create (_, vnode) {
+        bindFilter(vnode);
+    },
+    update (oldVnode, vnode) {
+        bindFilter(vnode);
+
+        let {data} = vnode;
+        let {data: oldData} = oldVnode;
+        let fid = data && data.f && data.f.fid;
+        let oldfid = oldData && oldData.f && oldData.
+        if (fid !== oldfid) {
+            unbindFilter(oldVnode);
+        }
+    },
+    destroy (vnode) {
+        unbindFilter(vnode);
+    }
+}; ⊘↵
```

Vue drive 案例: 实现过滤器

compiler

通过 vue-template-compiler 提供的 **transformNode**、**genData**

接口扩展 render 函数，实现在 VNode 上添加的属性和数据

```
157 module.exports = function mark(source, options) {
158     const {
159         ast,
160         render,
161         staticRenderFns,
162         errors
163     } = compileTemplate(source, {
164         preserveWhitespace: false,
165         modules: [
166             {
167                 transformNode: getMarkNode(options),
168                 postTransformNode: getPostTrans(options),
169                 genData: getGenData(options)
170             }
171         ]
172     });
173
```

```
123
124 function getGenData(options) {
125     let filterIdCounter = 0;
126     return function markNode(el, options) {
127         let filters = getFilters(el);
128         if (filters) {
129             const fid = filterIdCounter++;
130             el._fid = fid;
131             el._filters = filters;
132             let data = [
133                 `fid: ${fid}`,
134                 `t: [${filters.t.join(',')}]`,
135                 `p: ${JSON.stringify(filters.p)}`
136             ];
137             filters.vfor && data.push(`for: ${filters.vfor}`);
138             filters.velseif && data.push('if: true');
139             filters.vif && data.push('if: true');
140             const dataStr = `f: { ${data.join(', ')} },`;
141             return dataStr;
142         }
143         return '';
144     };
145 }
```

Vue drive 案例: directives

createAnimation API 会返回一个 animation 动画描述对象

通过 **v-animation 指令** 实现在 H5 上执行这个 **DOM 动画**

```
137
138  function initDirectives(Vue, directives = {}) {
139      Object.keys(directives).forEach(key => {
140          Vue.directive(key, directives[key]);
141      });
142  }
143
```

```
18  import {createAnimation, animationDirective} from './api/createAnimation';
19
20  export const directives = {
21      animation: animationDirective
22  };
23
```

```
289
290 /**
291  * 动画效果指令 v-animation
292  */
293 export const animationDirective = {
294     update(el, binding, vnode) {
295         const {value: animateCommands} = binding;
296         animationEffect(el, animateCommands);
297     }
298 };
```

快速开始

```
npm install @marsjs/cli -g

mars create mars-project

cd mars-project

mars serve -t swan,wx,h5
```

Thanks :)

Mars - Vue 驱动的多端开发框架

Github https://github.com/max-team/Mars

文档 https://max-team.github.io/Mars/