Vue源码剖析

课堂目标

Vue工作机制

Vue初始化流程

入口：src\platforms\web\entry-runtime-with-compiler.js

解析options，主要解析template选项

```
const { render, staticRenderFns } = compileToFunctions(template, {
        outputSourceRange: process.env.NODE_ENV !== 'production',
        shouldDecodeNewlines,
        shouldDecodeNewlinesForHref,
        delimiters: options.delimiters,
        comments: options.comments
      }, this)
      options.render = render
```

src\platforms\web\runtime\index.js

声明$mount

```
Vue.prototype.$mount = function (
  el?: string | Element,
  hydrating?: boolean
): Component {
  el = el && inBrowser ? query(el) : undefined
  // 执行挂载
  return mountComponent(this, el, hydrating)
}
```

src\core\index.js

初始化全局api

```
initGlobalAPI(Vue)
```

- initGlobalAPI

  ```
  Vue.set = set
  Vue.delete = del
  Vue.nextTick = nextTick
  initUse(Vue)
  initMixin(Vue)
  initExtend(Vue)
  initAssetRegisters(Vue)
  ```

src\core\instance\index.js

声明构造函数

```
function Vue (options) {
  this._init(options)
}

initMixin(Vue)
stateMixin(Vue)
eventsMixin(Vue)
lifecycleMixin(Vue)
renderMixin(Vue)
```

initMixin(Vue) src\core\instance\init.js

实现初始化函数_init

```
Vue.prototype._init = function (options?: Object) {
  initLifecycle(vm)
    initEvents(vm)
    initRender(vm)
    callHook(vm, 'beforeCreate')
    initInjections(vm) // resolve injections before data/props
    initState(vm)
    initProvide(vm) // resolve provide after data/props
    callHook(vm, 'created')
}
```

```
<div id="app">
  <h1>{{title}}</h1>
</div>

// app是Component实例
const app = new Vue({
  el: '#app',
  template: '<App/>' | app, // 如果设置template则以它为模板，否则以宿主元素为模板
  render: h => h(App) // 如果设置render，template直接忽略
}).$mount('#app')
```