版本2.5.13

从src/core/instance/index.js看起,

到src/core/instance/init.js,

1.采用flow对静态资源进行检查, 即

function initMixin (Vue: Class<Component>){...} 传参Vue 是Class类型

1. \_init函数
2. initMixin(Vue)
   1. /\*istanbul ignore if\*/ 让istanbul忽略下面的if代码块(istanbul <http://www.ruanyifeng.com/blog/2015/06/istanbul.html)>
   2. Mark是window.performance.mark 记录时间差,查看性能方面 在src/core/util/perf.js
   3. merge options mergeOptions函数 把new Vue()的参数对象中mixins，extends中的方法，属性绑到vm.$options上 vm就是new Vue() 在src/core/util/option.js
   4. initProxy(vm) 给vm弄个Proxy,有render.\_withStripped的话vm.\_renderProxy的hander是get，否是has 在src/core/instance/proxy.js
   5. initLifecycle(vm) 给vm对象加上 在src/core/instance/lifecycle.js
      1. vm.$parent = parent
      2. vm.$root = parent ? parent.$root : vm
      3. vm.$children = []
      4. vm.$refs = {}
      5. vm.\_watcher = null
      6. vm.\_inactive = null
      7. vm.\_directInactive = false
      8. vm.\_isMounted = false
      9. vm.\_isDestroyed = false
      10. vm.\_isBeingDestroyed = false
   6. initEvents(vm) 在src/core/instance/event.js一些实例方法
      1. vm.\_events = Object.create(null)
      2. 如果parent有实例方法updateComponentListeners(vm, listeners) vdom/helpers/update-listeners.js
      3. eventsMixin的时候在原型上添加下面方法
      4. target.$once(event, fn)
      5. target.$on(event, fn)
      6. target.$off(event, fn)
      7. 定义$emit()
   7. initRender(vm); 初始化Vdom上的节点，在src/core/instance/render.js
      1. vm.\_vnode = null // the root of the child tree
      2. vm.\_staticTrees = null // v-once cached trees
      3. vm.$slots = resolveSlots(options.\_renderChildren, renderContext)
      4. vm.$scopedSlots = emptyObject
      5. $attrs和$listeners用defineReactive 加的
      6. defineReactive 在src/core/observer/indexr.js
   8. callHook(vm, 'beforeCreate') 初始化'beforeCreate'事件
   9. initInjections(vm) // resolve injections before data/props
      1. 把vm.$options.inject的绑到vm上
   10. initState(vm)
       1. initProps(vm, opts.props)
       2. initMethods(vm, opts.methods)
       3. initData(vm)
          1. observe(data, true /\* asRootData \*/)
          2. ob = new Observer(value) core/observer/index.js
          3. 数组this.observeArray(value)=》循环observe(items[i]) 其他this.walk(value)=》循环defineReactive(obj, keys[i], obj[keys[i]])
       4. initComputed(vm, opts.computed)
       5. initWatch(vm, opts.watch)
   11. initProvide(vm) // resolve provide after data/props
       1. vm.\_provided = vm.$options.provide
   12. callHook(vm, 'created')初始化'Created'事件
   13. vm.$mount(vm.$options.el) 初始化渲染模板
3. stateMixin(Vue)
4. eventsMixin(Vue)
5. lifecycleMixin(Vue)
6. renderMixin(Vue)

Global

1. Index.js 在core/global-api/index.js 设置全局api
   1. Vue.set = set 因为有的时候 defineProperty的set不触发
   2. Vue.delete = del 同上
   3. Vue.nextTick = nextTick
   4. initUse(Vue) use插件
      1. Plugin是Object的时候plugin.install.apply(plugin, args) 或plugin是fun的时候plugin.apply(null, args)
   5. initMixin(Vue) mixin 混入一些行为
      1. mergeOptions(this.options, mixin)
   6. initExtend(Vue) 组件继承
      1. const Sub = function VueComponent (options) {
      2. this.\_init(options)
      3. }
      4. Sub.prototype = Object.create(Super.prototype)
      5. Sub.prototype.constructor = Sub
      6. Sub.cid = cid++
      7. Sub.options = mergeOptions(
      8. Super.options,
      9. extendOptions
      10. )
      11. Sub['super'] = Super
      12. 如果有Props proxy(Comp.prototype, `\_props`, key)
      13. 如果有Computed defineComputed(Comp.prototype, key, computed[key])
      14. Sub.extend = Super.extend
      15. Sub.mixin = Super.mixin
      16. Sub.use = Super.use
      17. ASSET\_TYPES.forEach(function (type) { //'component', 'directive','filter'
      18. Sub[type] = Super[type]
      19. })
   7. initAssetRegisters(Vue) 注册'component', 'directive','filter'
      1. if (type === 'component' && isPlainObject(definition)) { //'component'
      2. definition.name = definition.name || id
      3. definition = this.options.\_base.extend(definition)
      4. }
      5. if (type === 'directive' && typeof definition === 'function') { //'directive'
      6. definition = { bind: definition, update: definition }
      7. }
      8. this.options[type + 's'][id] = definition //filter

Util

1. shared/util 各种判断方法
2. Lang 三个方法isReserved（是否有\_和$） ，def(定义属性) ，parsePath（检查路径）
3. Env 浏览器区分和兼容, \_set类型
4. Options
5. Debug 发错的地方
6. Props
7. Error
8. next-tick