

《Vue基础》第十一单元

生命周期 / 组件通信



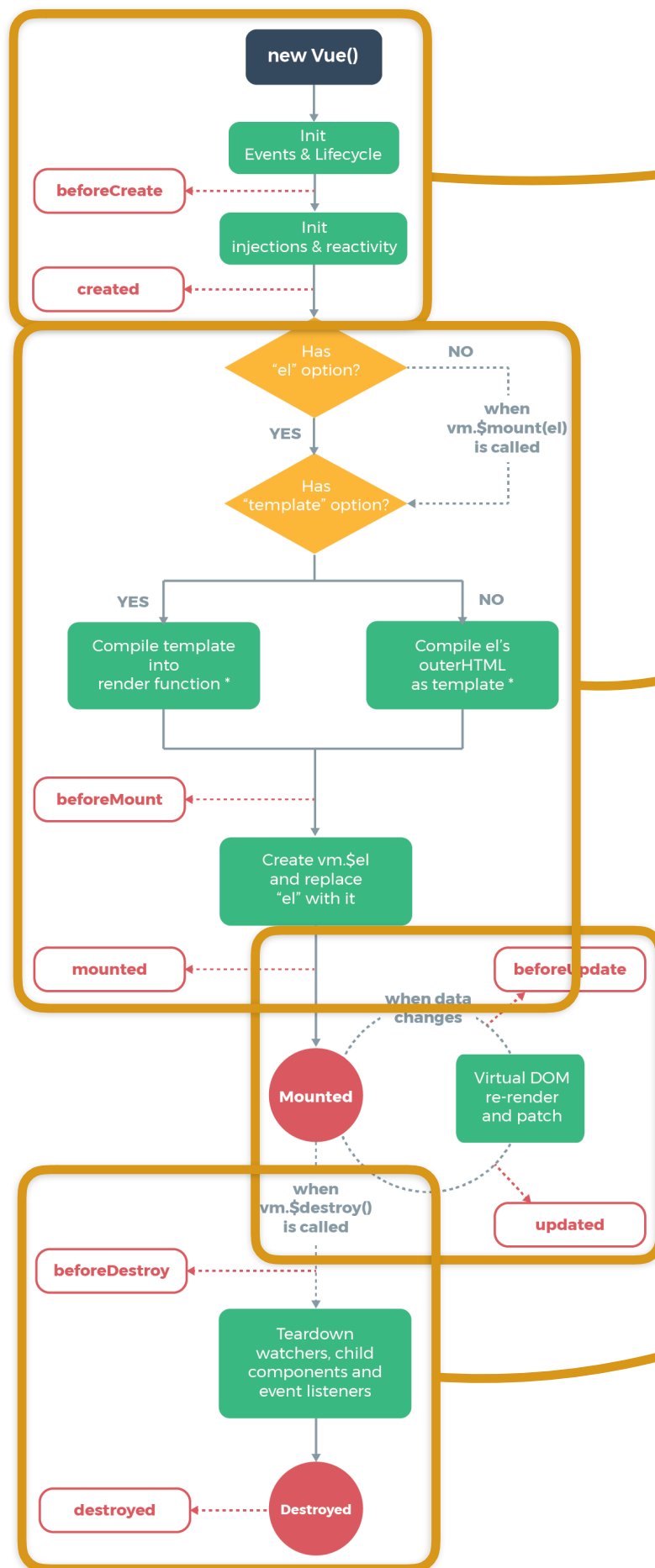
复习

- {{ }} / v-bind / v-html / v-model
- methods / v-on / computed / watch
- component / props / \$emit
- .vue / Pug / Sass
- props类型、默认值及校验 / 自定义v-model
- slot(name="***) / v-slot

生命周期

- Vue 实例被创建时要经历一系列的过程
- 过程中会运行一些钩子函数，给用户在不同阶段添加自己的代码的机会

钩子函数可以当作回调来理解,当系统执行到每个阶段时,检查是否有钩子,有则调用



1、创建

- beforeCreate
- created

2、挂载

- beforeMount
- mounted

3、更新

- beforeUpdate
- updated

4、销毁

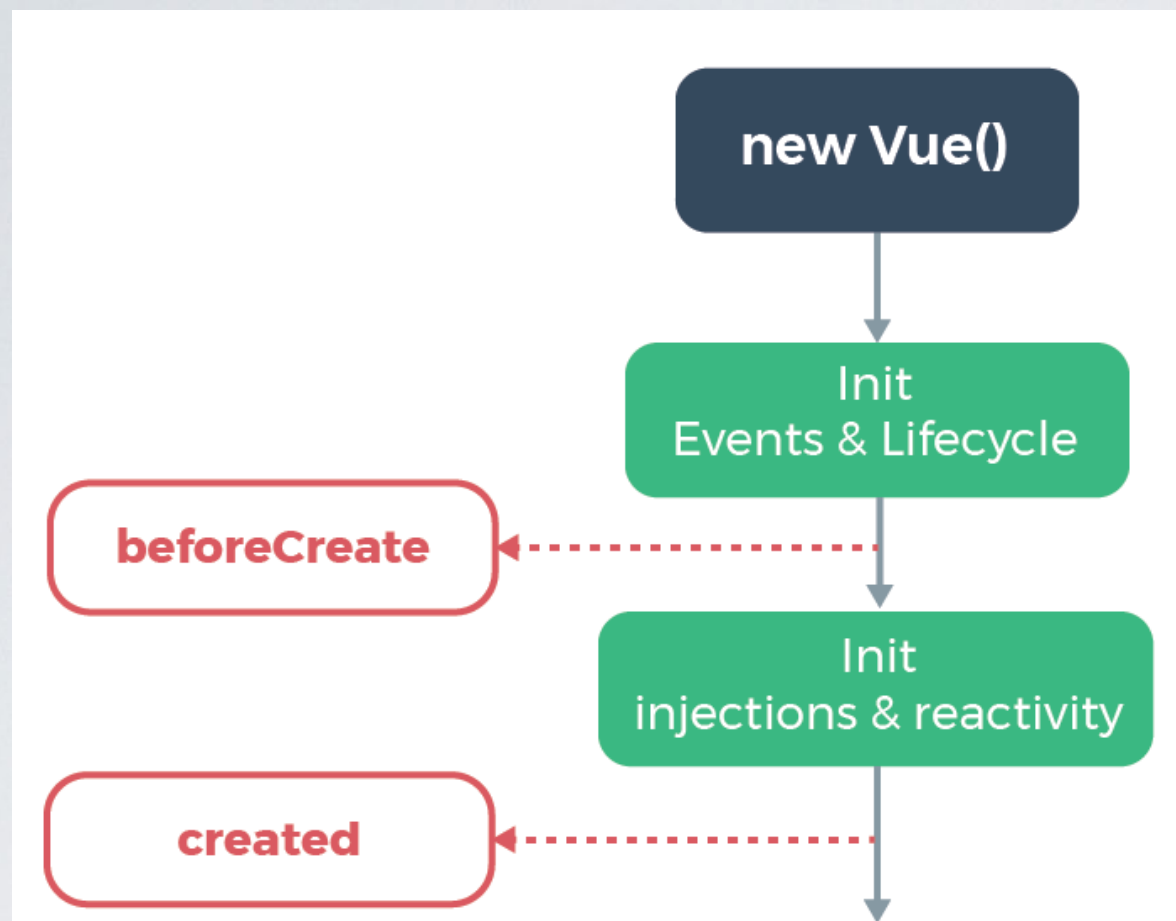
- beforeDestroy
- destroyed

- 激活 / 停用

- activated

- deactivated

* template compilation is performed ahead-of-time if using a build step, e.g. single-file components



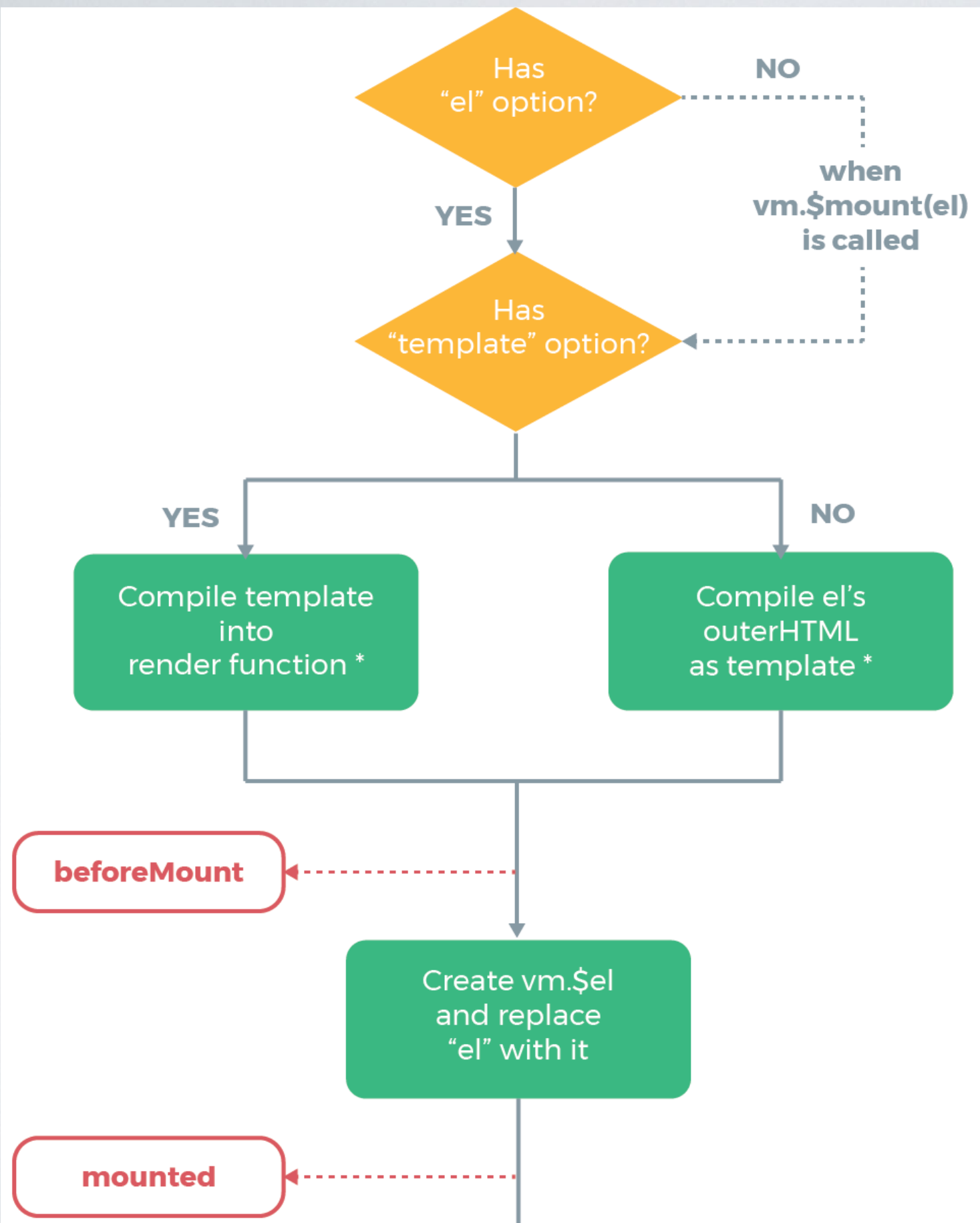
创建新的 Vue 实例

空 Vue 实例对象，只有默认的生命周期函数和事件

初始化响应式数据和方法

- 创建
- beforeCreate
- **created**

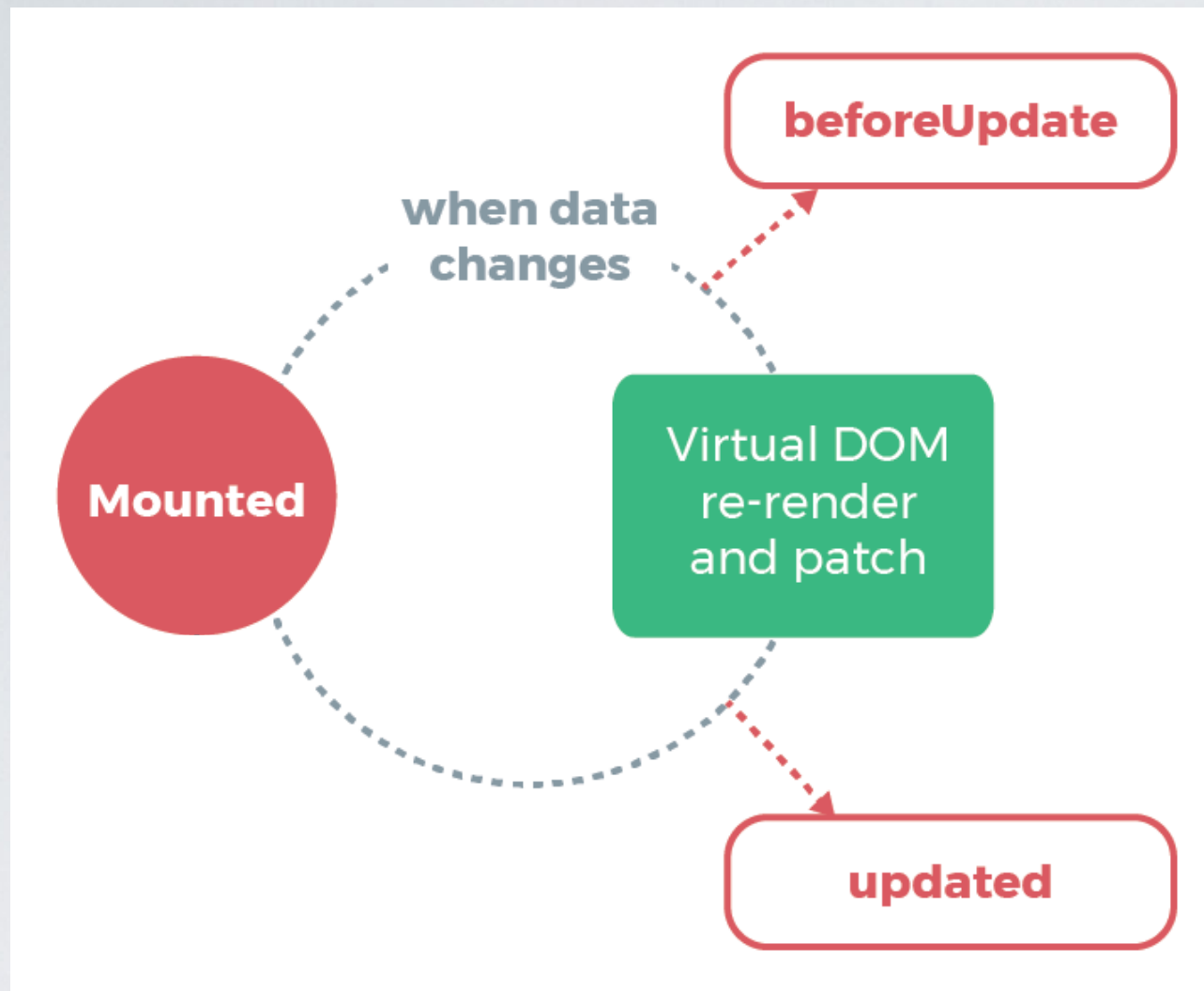
在这里调用后端接口取数据



- 挂载
- beforeMount
- **mounted** DOM

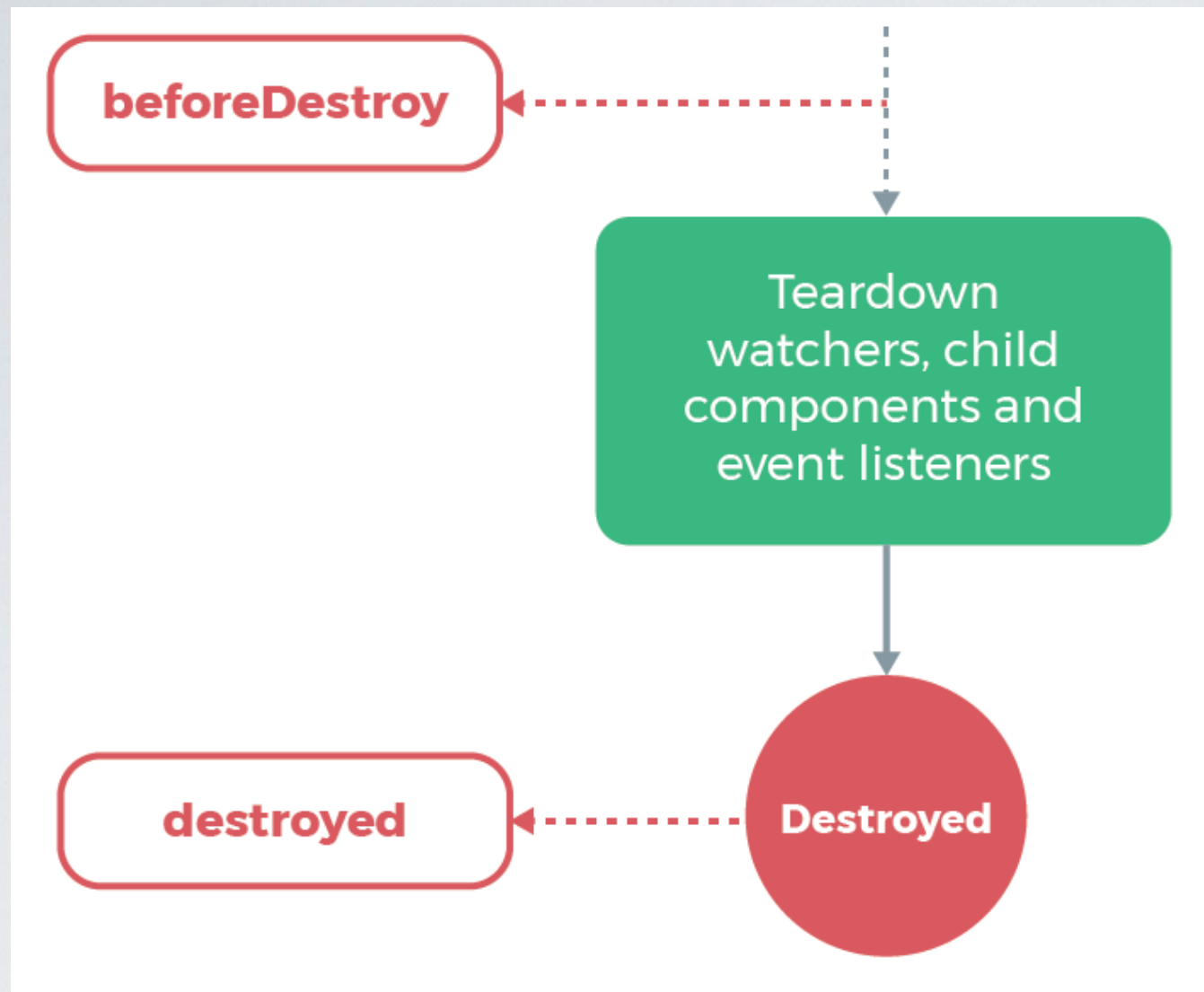
将模板编译成render函数

挂载替换模板，
生成可用HTML



实时监控data数据变化，重新渲染DOM

- 更新
- beforeUpdate
- updated



销毁数据、组件、事件
释放资源

- 销毁
- **beforeDestroy**
- destroyed

在这里销毁三方库，
解绑自定义事件

案例

课堂案例：生命周期演示

Father: 0
showChildOne
ChildOne: 0
hideChildTwo
ChildTwo: 0

[HMR] Waiting for update signal from WDS...

09:41:063 Father beforeCreate

09:41:073 Father created

09:41:073 Father beforeMount

09:41:079 ChildTwo beforeCreate

09:41:080 ChildTwo created

09:41:080 ChildTwo beforeMount

09:41:083 ChildTwo mounted

09:41:083 ChildTwo activated

09:41:084 Father mounted

10:11:377 Father beforeUpdate

10:11:380 ChildOne beforeCreate

10:11:380 ChildOne created

10:11:380 ChildOne beforeMount

10:11:381 ChildOne mounted

10:11:382 Father updated

>

混入

- 用 **mixin** 定义 Vue 组件的可复用功能
- 混入对象可以包含组件的任意选项
- 同名钩子函数都将被调用，混入的钩子先调用
- 值为对象的选项会合并，键名冲突时，以组件为准

混入

```
const mixin = {  
  data() {  
    return {  
      message: 'hello mixin'  
    }  
  },  
  methods: {  
    init() {  
      console.log('mixin init');  
    }  
  },  
  mounted() {  
    this.init();  
    console.log('mixin mounted');  
  }  
};
```

```
new Vue({  
  mixins: [mixin],  
  data() {  
    return {  
      message: 'hello component'  
    }  
  },  
  methods: {  
    init() {  
      console.log('component init');  
    }  
  },  
  mounted() {  
    this.init();  
    console.log('component mounted');  
  }  
}).$mount('#app');
```


组件缓存

问题思考：
keep-alive 和
v-show 的区别

- 用 **keep-alive** 缓存不活动的组件实例
- 激活 activated / 停用 deactivated

```
• keep-alive
  |
  | • component-a(v-if="showA")
  | • component-b(v-else)
```

课堂案例：缓存组件生命周期演示

组件通信

- 父子之间

\$emit、props传入函数、

\$root、\$parent、\$children、\$refs

- 兄弟之间

eventBus、\$on、\$emit

课堂案例：
妈妈喊你回家吃饭

```
window.eventBus = new Vue();
```

```
window.eventBus.$on('bus-event', val => {  
  console.log(val);  
});
```

```
window.eventBus.$emit('bus-event', 'Hi');
```

案例

课堂案例：MINI播放器和播放列表组件通信



总结

- 生命周期 Create、Mount、Update、Destroy
- 组件缓存 keep-alive , activated / deactivated
- 混入 mixins
- 组件通信 , 父子、兄弟

课后作业 1



- 编写全屏播放器组件
- 全屏播放器和MINI播放器共用 mixins
- MINI播放器、全屏播放器、播放列表三组件间相互通信



课后作业 2

- 了解 \$attrs / \$listeners
- 了解 provide / inject
- 了解 \$nextTick