

# REACT NATIVE 101

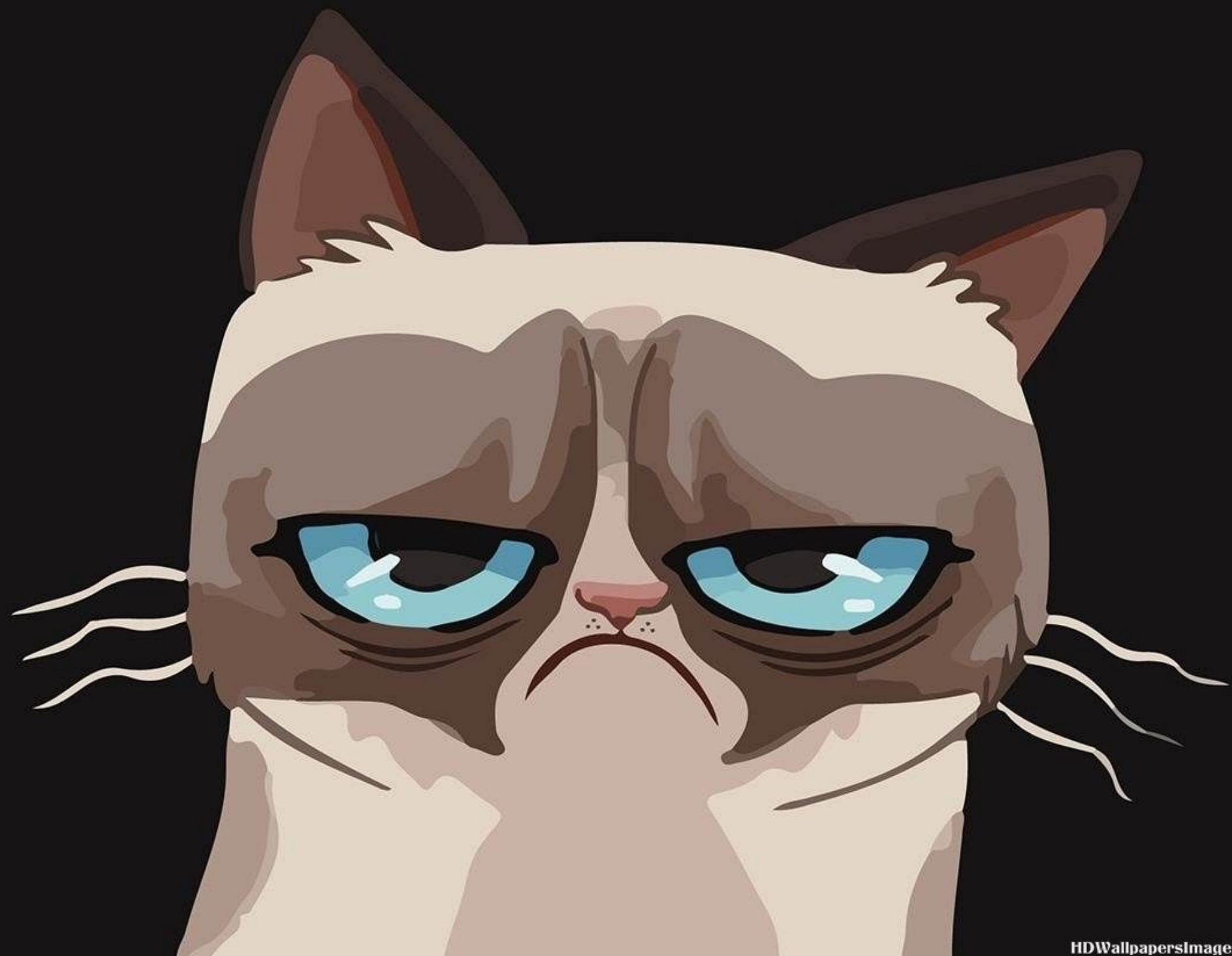
*"Inspirado"* EN LA CHARLA EN UIKONF DE NICK LOCKWOOD

¿QUÉ ES REACT NATIVE?  
UN FRAMEWORK PARA  
ESCRIBIR APPS *nativas*

¡GRACIAS FACEBOOK!

**¡DESARROLLAR APPS  
NATIVAS YA ES IDEAL!**

**No.**



# DESARROLLAR APPS EN 2015

1. **Ciclo de desarrollo lento**
2. **Ciclo de release eterno**
3. **Equipos separados por plataforma**
4. **APIs distintas para hacer lo mismo**

**¡USEMOS WEBVIEWS!**

A fluffy, light-colored cat with a white chest patch is shown in a dark room, looking startled with wide eyes. The cat is standing on a dark surface, possibly a bed or couch. The background is dark and indistinct.

**NO NO NO NO NO NO**

# WEBVIEWS EN 2015

1. Rendimiento **pobre**
2. Experiencia de usuario **inferior**
3. Funcionalidades **nativas limitadas**



# WEBVIEWS EN 2015

1. RENDIMIENTO POBRE

2. EXPERIENCIA DE USUARIO INFERIOR

3. FUNCIONALIDADES NATIVAS LIMITADAS



**REACT NATIVE**

**MISMA EXPERIENCIA DE USUARIO**  
**MEJOR EXPERIENCIA DE DESARROLLO**

# REACT NATIVE EN 2015

1. ~~Ciclo de desarrollo lento~~ Recarga instantánea
2. ~~Ciclo de release lento~~ Actualizaciones OTA legales !!!
3. ~~Equipos separados por plataforma~~ Mismas herramientas
4. ~~APIs distintas para hacer lo mismo~~ APIs consistentes

**WRITE ONCE, RUN ANYWHERE**

**ESCRIBE UNA VEZ, EJECUTA EN CUALQUIER LADO**

~~WRITE ONCE, RUN ANYWHERE~~  
~~WRITE ONCE, RUN ANYWHERE~~

~~ESCRIBE UNA VEZ, EJECUTA EN CUALQUIER LADO~~  
~~ESCRIBE UNA VEZ, EJECUTA EN CUALQUIER LADO~~

**DIFERENTES PLATAFORMAS  
TIENEN DIFERENTES  
EXPERIENCIAS DE USUARIO**

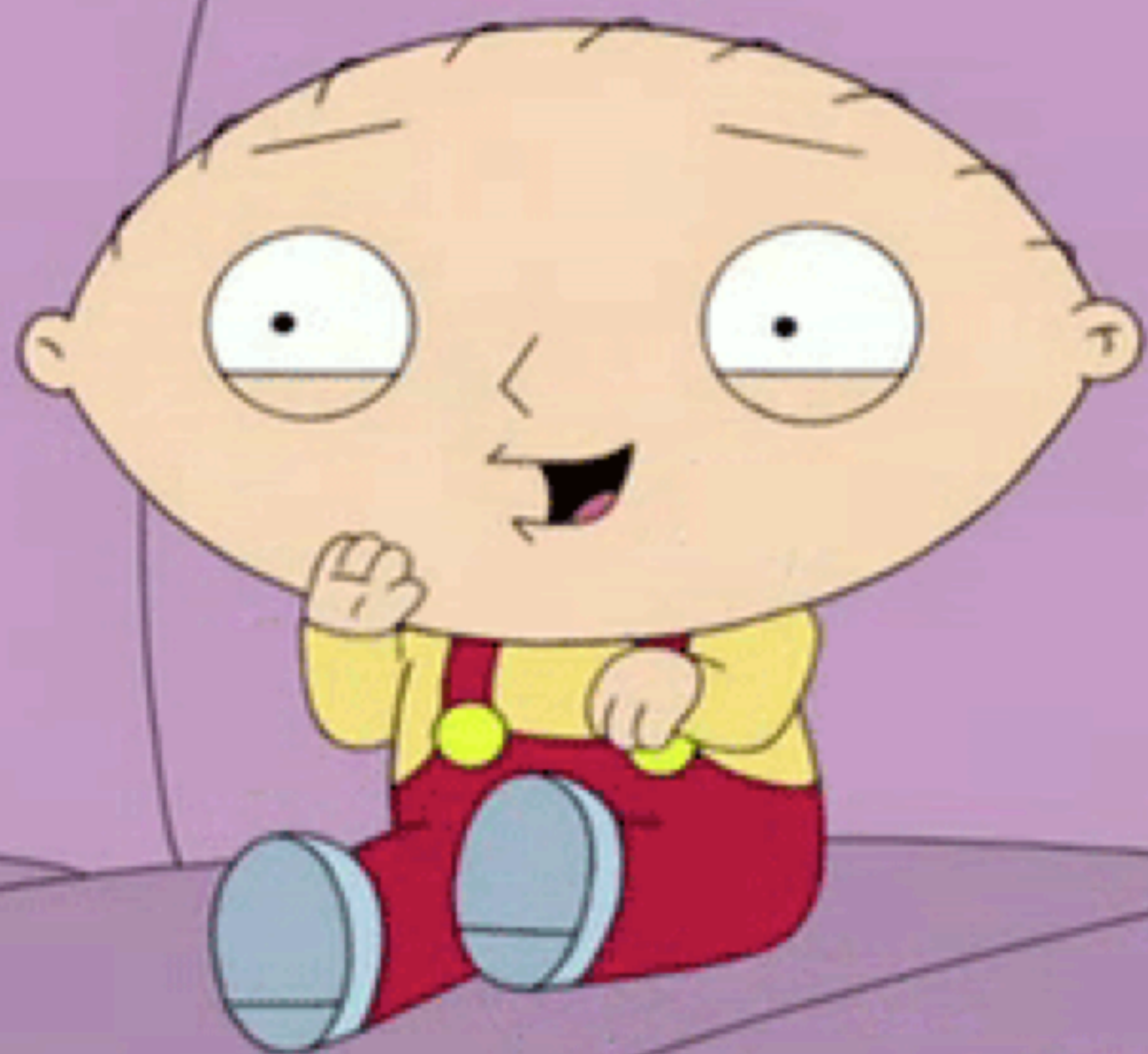
**LEARN ONCE, RUN ANYWHERE**

**APRENDE UNA VEZ, EJECUTA EN CUALQUIER LADO**



# REACT NATIVE EN 2015

1. ~~Rendimiento pobre~~ **Árbol de vistas virtual**
2. ~~Experiencia de usuario inferior~~ **Vistas nativas**
3. ~~Funcionalidades nativas limitadas~~ **Fácil de extender**



# APPS EN 2015

- ▶ **Hacer aplicaciones dinámicas es complicado**
  - ▶ **Pocos estados -> muchas transiciones**
- ▶ **Escalar aplicaciones web estáticas es mucho más fácil**
  - ▶ **Petición HTTP -> App -> Respuesta HTML**

**INTERFACES CON  
ESTADO  
MUTABLES**

# DIFÍCIL DE:

1. Entender
2. Debuggear
3. Mantener
4. Reutilizar
5. Testear

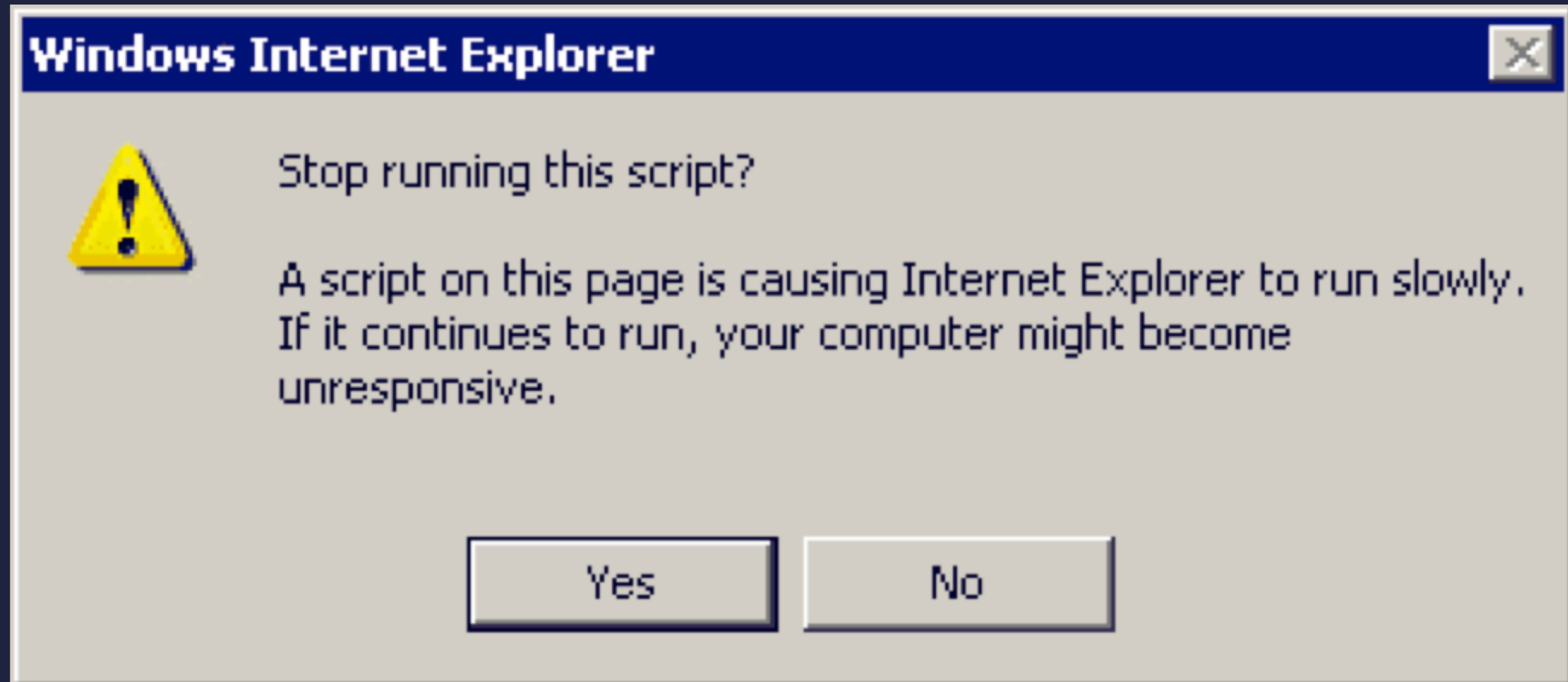
~~INTERFACES CON ESTADO MUTABLES~~

VISTAS IMMUTABLES

FUNCIÓNES DE RENDERIZADO PURAS

FLUJO DE DATOS EN UN SENTIDO

# AVISO: JAVASCRIPT



```
render: function(state) {  
  return (  
    <View name={state.name} />  
  );  
}
```



- ▶ **UI funcional basado en componentes**
  - ▶ **Árbol de vistas virtual**
- ▶ **Frontend en JavaScript (ES6, npm, TypeScript...)**
  - ▶ **Layout con JSX**
  - ▶ **Estilos inline con Flexbox (Next-Gen CSS)**
- ▶ **Backend nativo (UIKit, network, storage, location...)**

# ¿CÓMO FUNCIONA?

- ▶ **Hilo en segundo plano ejecuta nuestro JavaScript**
  - ▶ **Hilo principal habla con UIKit, red, etc**
  - ▶ **Comunicación mínima entre ellos**
  - ▶ **El hilo principal avisa de eventos**
- ▶ **JavaScript manda diffs de cambios en vistas**

# ¿DEBERÍA USARLO?

- ▶ **Muy joven, no es 100% estable/completo**
  - ▶ **Comunidad muy activa**
- ▶ **Un par de apps de Facebook ya en la AppStore**
  - ▶ **Disponible para Android pronto**
  - ▶ **Nuclide: IDE específico próximamente**

# ENLACES ÚTILES

- ▶ Documentación oficial ★
- ▶ Componentes de terceros
- ▶ React Native Playground



**¡A PROGRAMAR!**