Interfaz de usuario en dispositivos móviles

iOS, sesión 3: Autolayout

Puntos a tratar

- Motivación. Qué es *autolayout*
- Manejo de restricciones con el Interface Builder
- Manejo de restricciones por código

Autolayout es un sistema declarativo y basado en restricciones.

Las restricciones sirven para calcular automáticamente el *frame* de cada vista y adaptar la interfaz a las dimensiones reales de la ventana.

El único que domina el autolayout

Tipos de restricciones

- De alineación ("align")
  - centrar horizontal o verticalmente
  - alinear varios componentes entre sí
- De espaciado ("pin")
  - que haya determinado espacio entre componentes
  - que haya determinado espacio entre un componente y el borde izquierdo de la pantalla
  - Fijar el ancho de un componente
  - Que dos componentes tengan el mismo alto

2. Manipular restricciones visualmente

Manipular restricciones con Interface Builder

- Área de botones de la parte inferior derecha del *storyboard*
- Menú **`Editor`**
- Uso del ratón

Restricciones incorrectas

- **Insuficientes**: la posición y/o tamaño de algún elemento es ambigua
- **Contradictorias**: no se pueden satisfacer todas simultáneamente
- Usar `Resolve autolayout issues`

3. Más sobre las restricciones

Las restricciones formalmente

- Internamente, cada restricción es una ecuación lineal:

```
item1.atributo1 = multiplicador * item2.atributo2 + cte
```

- Algunas restricciones no son ecuaciones sino *inecuaciones*, sustituyendo el símbolo = por <= o >=.

Es decir, *autolayout* **está resolviendo un sistema de ecuaciones lineales** sujeto a restricciones

- Las propiedades de la restricción se pueden ver en el *size inspector*

Prioridades

- Cada restricción tiene asignada una **prioridad**, valor numérico que especifica su “importancia”
- El valor por defecto es 1000 -> la restricción **debe cumplirse**
- Valores<1000 -> se intentará cumplir la restricción pero es posible que no se cumpla si hay restricciones contradictorias de mayor prioridad.

"Prioridades" de los componentes

- A los componentes "no les gusta" ser "chafados" (*compression resistance*, valor alto por defecto)
- A los componentes "no les importa demasiado" evitar el *padding* (*content hugging*, valor bajo por defecto)

4. Formular restricciones usando código

A veces los elementos de la interfaz se crean dinámicamente y no se puede especificar el *layout* en Xcode. Otras veces queremos que cambien dinámicamente las restricciones para cambiar dinámicamente el *layout* o hacer animaciones

Métodos para formular una restricción con código

- Usar directamente el API de autolayout
- Usar el *Visual Format language* (recomendado frente al anterior, más intuitivo)

Ejemplo con el API: crear la restricción

```
superview.centerX = 1*button.centerX+0
```

```
NSLayoutConstraint *constraint = [NSLayoutConstraint
  constraintWithItem:button
  attribute:NSLayoutAttributeCenterX
  relatedBy:NSLayoutRelationEqual
  toItem:superview
  attribute:NSLayoutAttributeCenterX
  multiplier:1.0
  constant:0.0]
```

Ejemplo con el API: añadir la restricción a la vista

- Para que tenga efecto hay que añadirla a la vista con `addConstraint`

```
[self.miBoton.superview addConstraint:constraint];
```

Si son vistas “madre/hija” la añadiremos a la “madre”, y en otro caso *al ancestro común más cercano de ambas vistas*. Por ejemplo si fuera una relación entre dos botones dentro del mismo contenedor la añadiríamos al contenedor.

Visual Format language

"representación en modo texto" de la gráfica de las restricciones. El formato permite **representar un conjunto de restricciones con una cadena de caracteres**.

Ejemplo: separación estándar (8 pixels) entre el botón 1 y el 2

```
[boton1]-[boton2]
```

Más cadenas de formato

```
 [boton1]-20-[boton2] //separación de 20 puntos
- [boton1(50)]-20-[boton2(>=50)] //entre paréntesis el ancho
- [boton1]-20@800-[boton2] //prioridades con la @
- [boton1]-20-[boton2(==boton1)] //==, mismo tamaño
- V:[topField]-10-[bottomField]  //V -> *layout* en vertical
- |-[find]-[findNext]-[findField(>=20)]-| //Las barras son los bordes del contenedor
```

Ejemplo con código

```
NSLayoutConstraint *constraint =
 [NSLayoutConstraint
   constraintsWithVisualFormat:
   @"[cancelButton]-[acceptButton]"
   options: NSLayoutFormatDirectionLeadingToTrailing |
            NSLayoutFormatAlignAllCenterY
   metrics:nil   //para ctes. simbólicas en la restricción
   views:viewsDictionary];
```

```
UIButton *cancelButton = ...;
UIButton *acceptButton = ...;
NSDictionary *views = NSDictionaryOfVariableBindings(cancelButton,
                          acceptButton);
```