

# INTERACCION GRAFICA DE OBJETOS

Estudio, desarrollo y exploración hacia una interfaz activa de objetos gráficos.

Las interfaces de usuario (GUI) actuales basadas en ventanas y widgets estándar presentan grandes limitaciones para su construcción, modificación e interacción de usuarios y programadores. En una GUI estándar no es posible acceder e interactuar directamente con los objetos de la aplicación. Para construir y editar ventanas y widgets es necesario abrir editores externos a la vista observada. En la charla propuesta presento (mediante un prototipo experimental) herramientas y conceptos avanzados para la creación e interacción de nuevas interfaces activas que superan dichas limitaciones.

## **Motivaciones:**

Poder monitorear, comparar, inspeccionar y modificar cualquier objeto directamente en las propias ventanas de la aplicación o del ambiente de trabajo. Eliminar el uso de editores o inspectores externos (abiertos en nuevas ventanas) ajenos a la propia vista del objeto observado. Implementar el volcado activo de objetos (ej: MorphicWrappers) y eliminar el uso de editores para construir las vistas de la aplicación (ej: WindowBuilder).

Destruir el concepto de ventana como compartimentos cerrados. En su reemplazo producir un cuerpo simple de conceptos y usos gráficos independientes de widgets, ventanas y/o vistas windows. No debe haber límites gráficos ni de interfaz para acceder, comunicar e interactuar con y entre los objetos contenidos en las ventanas y/o en sus partes. Un ejemplo de ventanas cerradas incompatibles es cuando abrimos dos inspectores y queremos saber si los objetos inspeccionados (o algunos de sus colaboradores) son iguales.

Explorar la programación o indicación gráfica de acciones y visualización de resultados. Así como podemos volcar objetos en cualquier momento y lugar también podríamos volcar sus mensajes (acciones) que gráficamente aplicaríamos a uno o mas objetos vecinos o ubicados en otras ventanas. Por ejemplo cuando depuramos código es frecuente inspeccionar un objeto y necesitar saber cuando cambio alguna de sus variables o cuando cambio la respuesta a un mensaje dado. Lo que hacemos normalmente es avanzar un paso el debugging y verificar en el Inspector si cambio el objeto. También podemos agregar un halt de instancia en el objeto, etc. La idea es indicar la acción de “avanzar el <debugger> hasta que se cumpla <una condición> en <un inspector>” de manera gráfica y directa.

## **Indice temático de la exposición:**

1. Limitaciones del diseño, construcción e interacción de las interfaces de ventanas estándar.
  - 1.1. Como indicamos propiedades, estilos y comportamiento a las vistas, ventanas y widget de la aplicación.
  - 1.2. Como indicamos deformaciones y reubicaciones de paneles y widget (LayoutFrame, EnhancedLayoutFrame, ShutterPane).
  - 1.3. Agregado, uso y visualización estática y dinámica de paneles y widget.
  - 1.4. Inspección básica de paneles, widget y menús.
2. Demostración de la edición activa de widget.
3. Experimentación, aplicación y uso de nuevos conceptos de visualización e interacción gráfica de objetos.
  - 3.1. Distinción por proximidad.
  - 3.2. Despiece 2D.
  - 3.3. Conexión y uso de espacios gráficos embebidos o flotantes.
  - 3.4. Modo tablero de trabajo.
  - 3.5. Marcas y señales informativas e interactivas (ej: FramingMark).
  - 3.6. Monitor de mensajes y notificaciones windows.
  - 3.7. Transcript gráfico (modos cascada o continuo y monitor de historia acumulada o exposición).
4. Comentarios finales.