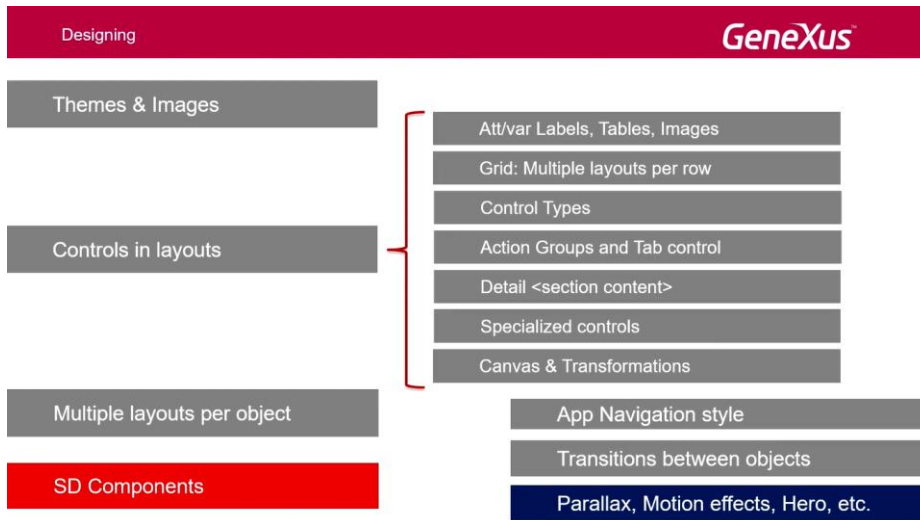


## SD Components



Veamos ahora cómo reutilizar componentes con los SD Components.

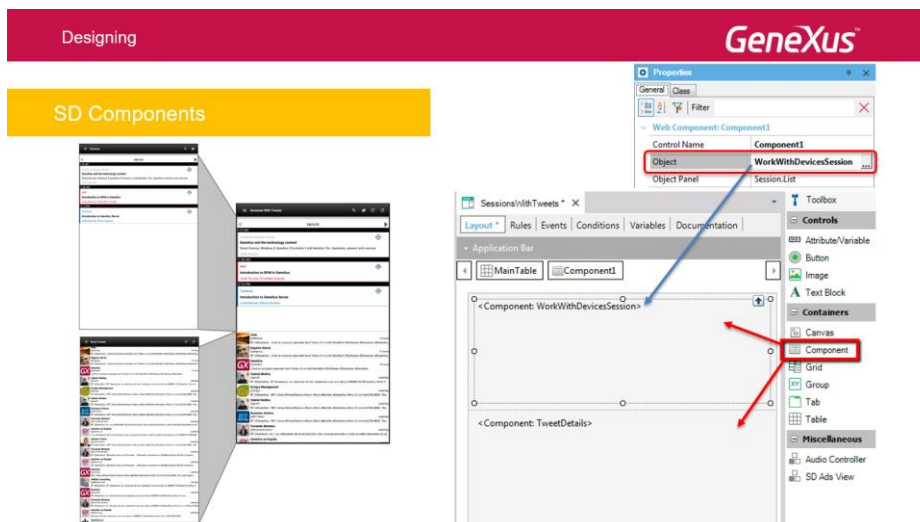
Los **SD Components** son controles que pueden contener a objetos del tipo Panel for Smart Devices o Work With for Smart Devices. Estos objetos ejecutan en el contenedor en forma independiente del mismo y sin perder sus características propias.

A su vez un Panel for Smart Devices o un Work With for Smart Devices puede contener uno o más SD Components, por lo cual se provee al desarrollador de un alto grado de reusabilidad. En este sentido, su funcionamiento es similar a los Web Components, respecto a los web panels.

Existen algunas restricciones a tomar en cuenta:

- 1) La propiedad Object, que permite asignar a un SD Components el nombre del objeto que ejecutará, sólo puede asignarse en tiempo de diseño.
- 2) Los SD Components no pueden incluirse dentro de un control Grid
- 3) La sincronización de los componentes en tiempo de ejecución se realiza mediante Eventos Globales.

Veremos esto a continuación.



Los eventos son acciones que disparan los usuarios y que pueden ser detectados por los programas, ejecutándose un código específico en respuesta a esas acciones. Por ese motivo, los eventos son usualmente locales al programa donde están definidos.

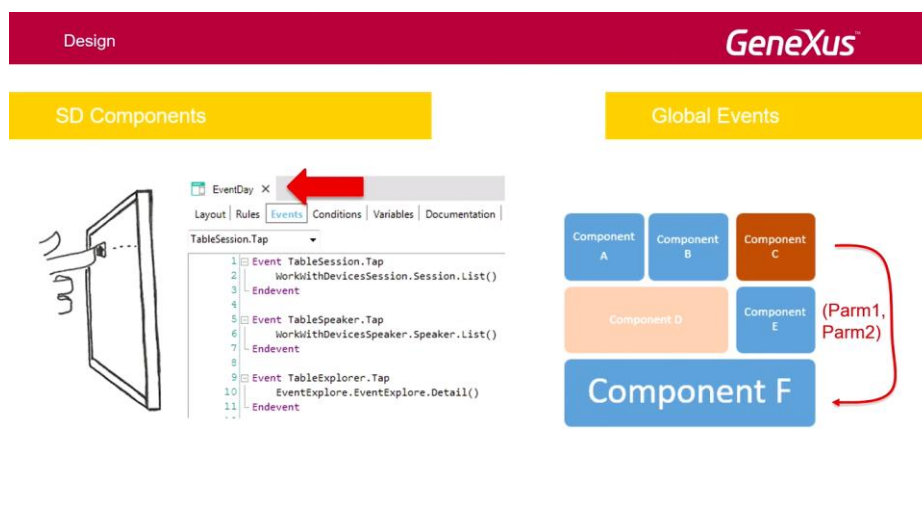
Los **Eventos Globales** en cambio, permiten comunicar varios componentes de una misma aplicación. A diferencia de los eventos locales que son iniciados por un usuario al interactuar con la aplicación, los Eventos Globales son un código que es disparado por el código de otro evento.

Por ejemplo puede darse el caso de que un usuario ingrese cierta información en un panel (por ejemplo ingresar una factura), y otro panel reacciones a esa acción, (por ejemplo imprimiendo esa factura).

Dicho de otra manera, un Evento Global puede ser definido en un cierto componente e invocado desde cualquier otro componente.

Los **Global Events** permiten sincronizar acciones entre SD Components y también tienen aplicación en ambiente Web, para sincronizar acciones entre Web Components. A diferencia de los eventos locales, los Global Events pueden recibir parámetros, lo que permite el intercambio de información entre componentes.

Esto hace que el uso de los Eventos Globales combinados con los componentes, minimicen el grado de dependencia entre componentes (o sea el acoplamiento), sin afectar la cohesión de cada componente.



Los Eventos Globales se definen en un objeto externo que podemos personalizar. El objeto externo se llama **GlobalEvents**.

Para crear eventos, nos posicionamos en el nodo Events y damos "Enter". Opcionalmente podemos agregarle parámetros al evento definido.

Una vez que los eventos están definidos, podemos invocarlos o capturarlos desde un objeto Panel o Work With for Smart Devices.

## SD Components

## Global Events: how to use them

The screenshot shows the GeneXus design environment. On the left, a tree view under 'Root Module' shows 'CommonApi' and 'GlobalEvents'. Below this, a diagram shows two 'SDPanel' components, each with a 'WorkWithDevices' icon, connected by a 'Trigger' arrow pointing to a 'Handling' arrow. In the center, the 'GlobalEvents' structure is displayed, showing a table with columns 'Structure', 'Type', 'Is Colle...', and 'Description'. The table lists events: 'AnEvent' (Type: None), 'AnEventParam' (Type: None), 'Numeric' (Type: Numeric(8,0)), and 'String' (Type: VarChar(40)). On the right, the 'Properties' window for 'ExternalObjectEvent: EventA(Param1,Param2)' is shown, with fields for 'Internal Name' (EventA), 'Description', 'Type' (Numeric), 'Length' (8), 'Decimals' (0), 'Signed' (False), 'Collection' (False), 'Based on' (none), 'XML Name', 'Is Static' (True), and 'External Member' (Default).

Veamos esto con un ejemplo.

Supongamos que tenemos dos SD Panels independientes llamados Sender y Receiver. La misión de Sender es enviar información al Receiver para actualizar datos.

En la tabla de arriba vemos cómo interaccionan ambos paneles: el primero envía una señal y el segundo cuenta cuántas veces recibe la señal del primero. También, el primero, el Sender, envía un dato numérico y una cadena al segundo, y el segundo despliega ese dato numérico y esa cadena recibida.

Finalmente, creamos un SD Panel que contendrá 2 SD Components, uno asociado al Sender y otro asociado al Receiver, de forma que podemos ver la interacción de los paneles en una única pantalla.

## Global Events: how to use them

The screenshot shows the GeneXus design environment. On the left, the 'GlobalEvents' structure is displayed, showing a table with columns 'Structure', 'Type', 'Is Colle...', and 'Description'. The table lists events: 'AnEvent' (Type: None), 'AnEventParam' (Type: None), 'Numeric' (Type: Numeric(8,0)), and 'String' (Type: VarChar(40)). Below this, a diagram shows two 'SDPanel' components, each with a 'WorkWithDevices' icon, connected by a 'Trigger' arrow pointing to a 'Handling' arrow. In the center, a table shows the interaction between 'Receiver' and 'Sender' components. The table has columns 'Receiver' and 'Sender'. The 'Receiver' column contains: 'Show how many times it receives the signal', 'Display the last numeric and string received', and a diagram showing a 'Count' box with '&Count' and a 'Last data received' box with 'Numeric (&Numeric)' and 'String (&String)'. The 'Sender' column contains: 'Send a signal', 'Send a numeric and string value', and a diagram showing a 'Send signal' box and a 'Send values' box with 'Numeric (&Numeric)' and 'String (&String)'. Below the table, a diagram shows two 'SDPanel' components, each with a 'WorkWithDevices' icon, connected by a 'Trigger' arrow pointing to a 'Handling' arrow. The 'Sender' component is labeled '<Component: Sender>' and the 'Receiver' component is labeled '<Component: Receiver>'.