

Funciones

Usar múltiples controles en la fórmula

Es posible hacer referencia a varios controles en la misma fórmula para proporcionar resultados dinámicos a los usuarios de su aplicación.

En el siguiente ejemplo, explicamos cómo puede crear una experiencia de formulario basada en varias entradas, que genera un mensaje como comentarios para el usuario. Nuestro ejemplo muestra una forma de calcular el coste de una cantidad concreta de un artículo en particular y mostrarlo como un mensaje con información monetaria incluida.

Nota

Mientras sigue estos pasos y la fórmula devuelve un error, recuerde que la configuración de idioma del entorno (o localización) de Power Apps puede influir en algunos separadores y operadores. Por ejemplo, la fórmula `Text(ThisItem.Price, "$ ##.00")` se expresa en un idioma y una región que utilizan un punto como separador de decimales como, por ejemplo, Japón o el Reino Unido. Sin embargo, para un idioma y una región donde se usa una coma como separador decimal, como Francia o España, la fórmula sería: `Text(ThisItem.Price; "$ ##,00")`.

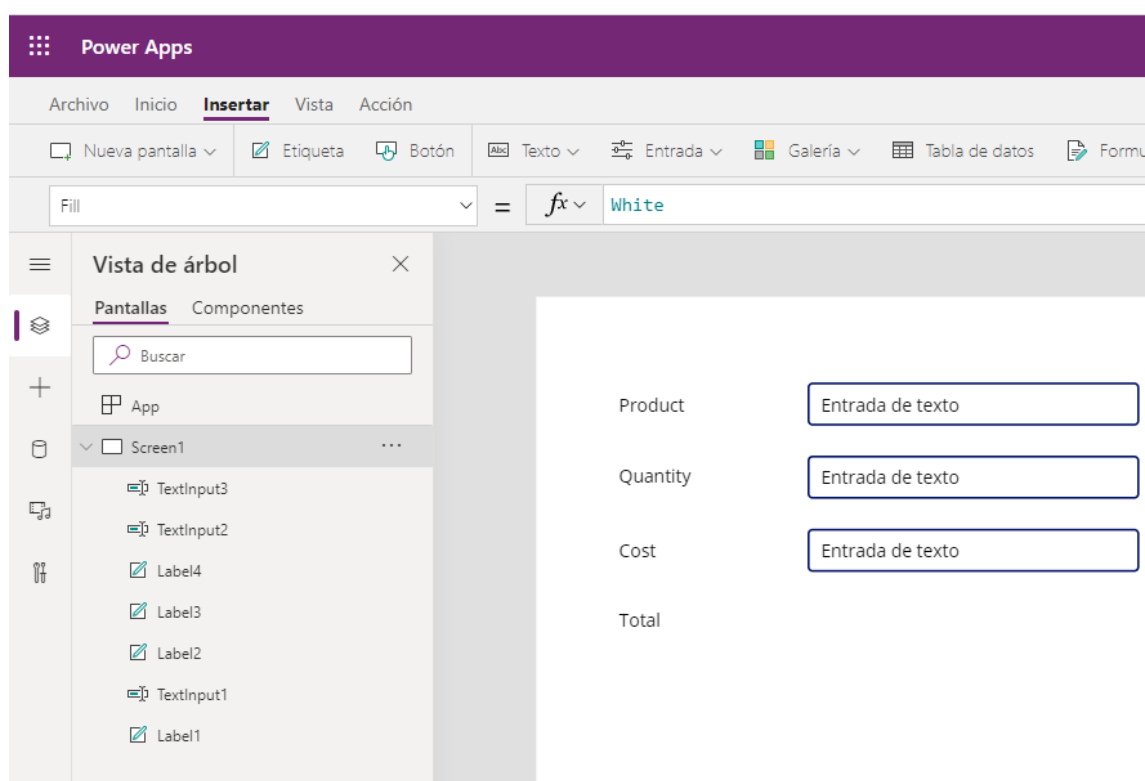
El operador de selección de propiedad (punto) en `ThisItem.Price` es siempre el mismo, sin importar cuál sea el separador decimal, pero observe que el separador decimal y el separador de la operación de encadenamiento cambian a coma y punto y coma respectivamente. Internamente, la fórmula no cambia, lo que cambia es cómo la muestra y edita el autor.

1. En una instancia o pestaña separada del navegador, vaya a make.powerapps.com y cree una nueva aplicación (puede usar la misma aplicación que creó en otra unidad).
2. Seleccione **+ Insertar > Etiqueta de texto**.
3. Cambie la propiedad **Text** por "Product".
4. Agregue tres etiquetas de texto más y colóquelas debajo de su primera etiqueta. Cambie las propiedades de **Text** "Quantity", "Cost" y "Total", como la siguiente imagen.

Sugerencia

Puede cambiar el nombre de la propiedad Text de cada etiqueta haciendo doble clic en el propio control. Al hacer doble clic, se resalta el texto existente en el control y luego, al escribir el texto, se inserta automáticamente el texto en su control de etiqueta (también actualiza su fórmula Power *fx*, incluidas las comillas).

5. A continuación, inserte controles **Text Input** en la aplicación de lienzo y ordénelos como se muestra en la siguiente captura de pantalla.

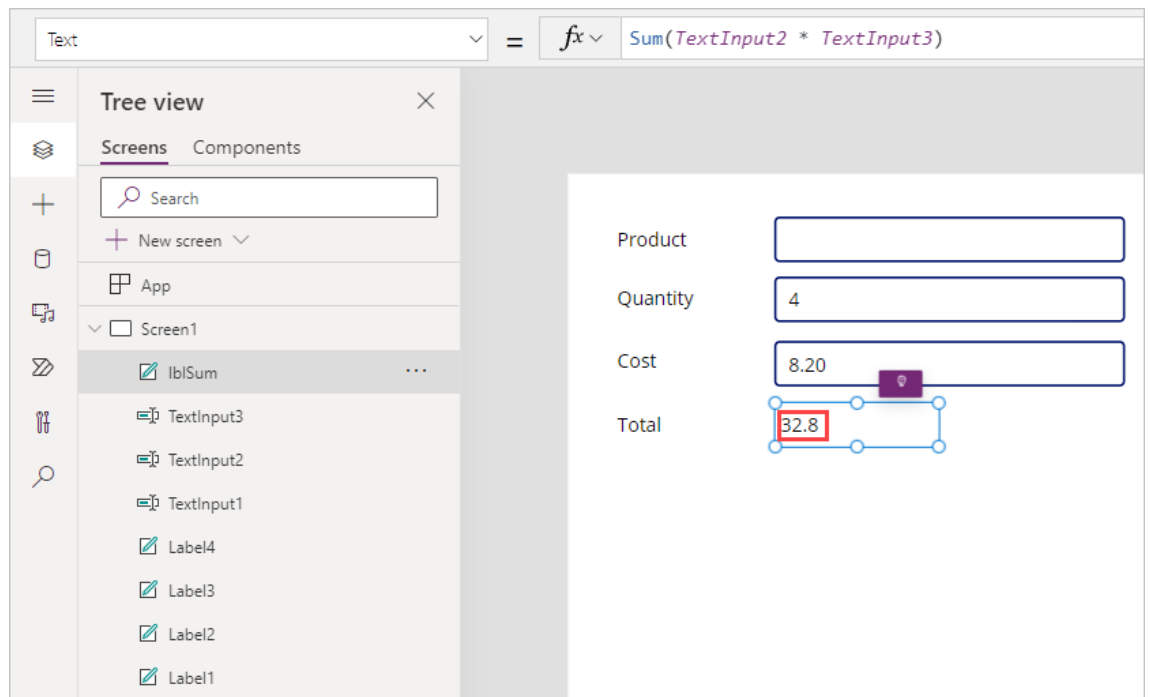


6. En la propiedad **Default** de cada **TextInput**, elimine "Entrada de texto" para que cada **Default** quede en blanco.
7. Inserte otra **Etiqueta de texto** a la derecha de *Total*. Cambie el nombre de este control por *lblSum*.
8. Cambie la propiedad **Text** por lo siguiente:

`Sum(TextInput2 * TextInput3)`

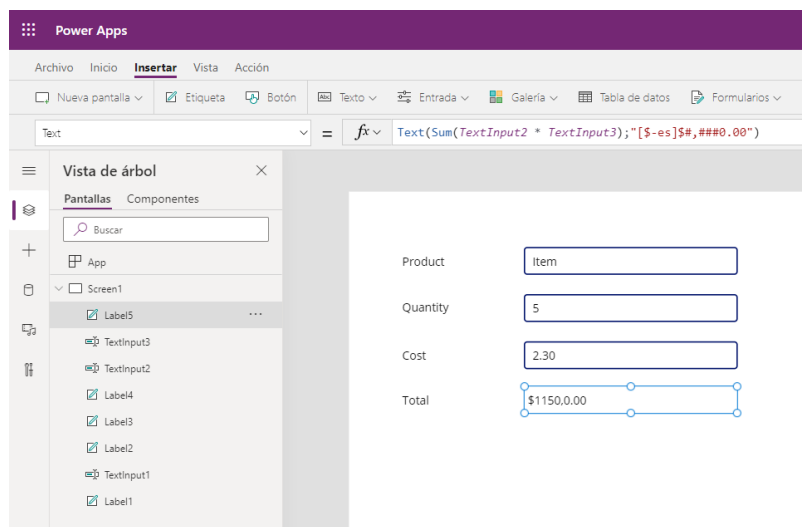
(En este ejemplo, TextInput2 es la cantidad comprada y TextInput3 es el coste por producto).

9. Ahora obtenga una versión preliminar de la aplicación seleccionando el botón **Reproducir** situado en la esquina superior derecha. Pruebe la fórmula introduciendo 4 para el campo de entrada **Cantidad** y 8.20 para el **Coste**. Observe que actualmente no mostramos dos decimales (lo cual es normal cuando se representa una divisa) o un signo de divisa, así que modificamos la fórmula para mostrarlos a continuación.



10. Para modificar la fórmula para que muestre el valor como *Divisa*, actualice la propiedad **Text** de **lblSum** para:

`Text(Sum(TextInput2 * TextInput3), "$#,###0.00")`



Observe cómo ahora *Total* muestra el resultado como una *Divisa*.

¿Qué acaba de suceder? Le dijimos a nuestra fórmula que agregara un signo de dólar al principio de una fórmula que incluyera una coma en el lugar de los miles, y que pusiera un cero delante de un decimal si el valor era menor que 1, y que agregara dos lugares después del decimal, incluso si el valor era cero.

11. Llevemos nuestra fórmula un paso más allá agregando un resumen de transacción. Agregue una última **Etiqueta de texto** bajo los demás controles y estire el ancho para que ocupe al menos la mitad del lienzo de la pantalla. En la propiedad **Text**, introduzca la fórmula siguiente:

```
"Your "& TextInput1.Text & " total is: " & lblSum.Text
```

Donde *TextInput1* es el control de entrada de texto junto a **Producto**

12. Ponga la aplicación en modo de versión preliminar e introduzca un valor en el control de la entrada de texto, junto a **Producto**. Observe cómo obtiene un mensaje de resumen de transacción que incorpora los valores de todos los campos de entrada. En nuestro ejemplo, hemos introducido *Widget* como nombre del producto. Introduzca valores diferentes en los campos de entrada de texto para ver cómo el total y el mensaje de resumen de transacción van cambiando dinámicamente.

| | |
|--|-------------------------------------|
| Product | <input type="text" value="Widget"/> |
| Quantity | <input type="text" value="4"/> |
| Cost | <input type="text" value="8.20"/> |
| Total | \$32.80 |
| <div>Your Widget total is: \$32.80</div> | |

Ahora hemos usado tres controles de entrada con la fórmula **Sum** para calcular un total y mostrar un mensaje de resumen de transacción. Con suerte, puede ver el potencial de usar múltiples controles en fórmulas para proporcionar comentarios dinámicos y relevantes a los usuarios de su aplicación.

A continuación, presentamos cómo podemos usar un control para modificar las propiedades de otros controles, específicamente cómo podemos cambiar la posición de visualización de un control. Por lo tanto, mantenga abierta la aplicación de práctica.

Usar entradas para ajustar las posiciones de un control

Vamos a usar un ejemplo para entender el posicionamiento de un control en el lienzo y cómo usar la información que especifique el usuario para modificarlo. Como introducción, podemos decir que la ubicación de un control en el lienzo se basa en una combinación de dos propiedades: las propiedades **X** e **Y**.

- **X**: distancia entre el borde izquierdo de un control y el borde izquierdo de la pantalla.
- **Y**: distancia entre el borde superior de un control y el borde superior de la pantalla.

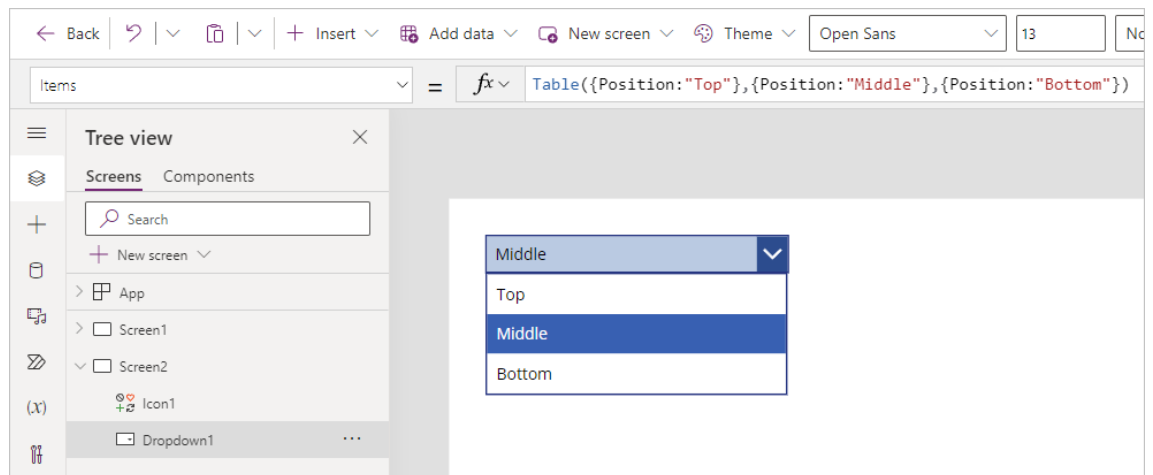
Recuerde que las propiedades **X** e **Y** se aplican a todos los controles, excepto las pantallas. Ahora vaya a su aplicación y aprendamos con la práctica.

1. Primero vamos a crear una nueva pantalla para la aplicación seleccionando + **Nueva pantalla** en el panel **Vista de árbol**, luego seleccione **En blanco**.
2. A continuación, Insertemos un control dropdown. Seleccione el botón + **Insertar** y escriba *desplegable* en el campo de búsqueda, luego seleccione **Desplegable**.
3. Cambie la propiedad **Items** de la lista desplegable de **DropDownSample** a lo siguiente:

Fórmula de PowerAppsCopiar

```
Table({Position:"Top"},{Position:"Middle"},{Position:"Bottom"})
```

Esto crea una tabla de registros con una única columna denominada **Position** con tres registros denominados *Top*, *Bottom* y *Middle*. Si selecciona el control dropdown mientras mantiene presionada la tecla **Alt**, puede ver que se muestran estos valores.

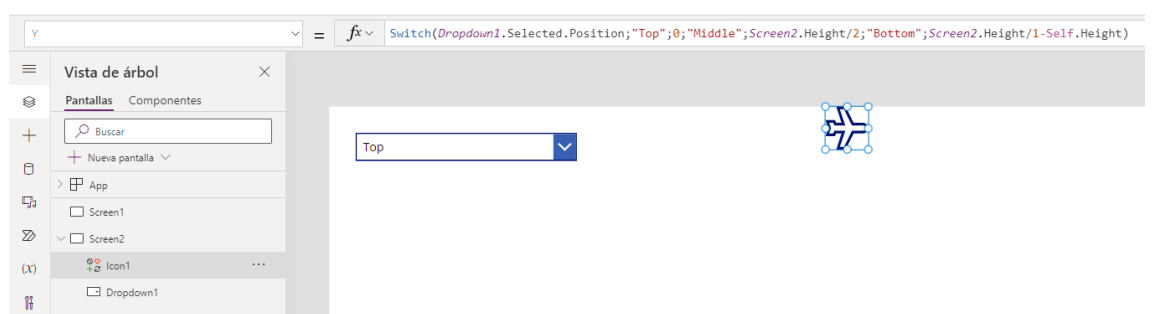


4. A continuación, agregamos un control que podemos reposicionar dinámicamente en la pantalla. Una vez más, seleccione **+ Insertar** desde la cinta, escriba *avión* en el campo de búsqueda y seleccione el icono del **avión**. Arrastre el icono del **avión** hasta el centro de la pantalla.
5. Ahora hagamos que el icono del **avión** aparezca a diferentes alturas de la pantalla basándonos en el control dropdown. Para ello, seleccione el icono de avión y busque la propiedad **Y**. Observe que actualmente tiene un valor de número estático. Cambie el valor de la propiedad **Y** por el siguiente valor:

Switch(Dropdown1.Selected.Position,"Top",0,"Middle",Screen2.Height/2,"Bottom",Screen2.Height/1-Self.Height)

Nota

El control dropdown puede tener un nombre que no sea *Dropdown1*. Reemplace *Dropdown1* por el nombre específico de su control.



6. Para divertirnos un poco (y probar la fórmula), ponga la aplicación en modo de versión preliminar. Ahora seleccione cada posición en el menú desplegable. Observe que el icono **Avión** ajusta su "altura" (o coordenada **Y**) a medida que cambia el valor desplegable.

Explicación

Hemos podido lograr el movimiento de nuestro icono de avión porque nuestra propiedad **Y** es solo un valor numérico. Sin embargo, no todos los números funcionarían, por lo que tuvimos que asignar números que mostrarían nuestro icono en la pantalla.

Usamos una función **Switch** basada en el valor de posición de nuestro control desplegable, asignando un valor diferente a **Y** basado en la entrada del usuario. El cero está en la parte superior y toda la altura de la pantalla está en la parte inferior. Usamos la propiedad **Height** de nuestro control de pantalla para dar cuenta de cualquier plataforma que utilice esta aplicación.

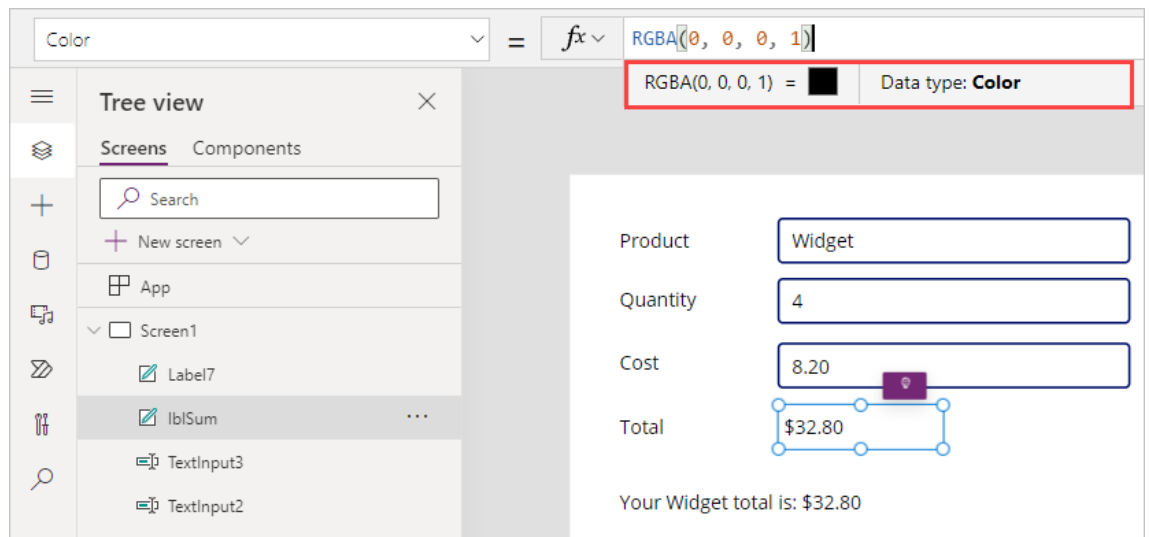
También usamos la propiedad **Height** del icono del avión (Self.Height) para el valor "Bottom" para que pudiéramos ver el icono del avión en la parte inferior de la pantalla; de lo contrario, habría estado fuera de nuestra pantalla visible.

Puede cambiar las propiedades de sus controles para modificar el comportamiento de otros. Si lo hace, le brindará una excelente manera de establecer la experiencia de usuario. Mantenga abierta la aplicación de práctica y, en el siguiente módulo, aprenderemos a mejorarla aún más aplicando formato condicional.

Formato condicional del color del texto

Siguiendo con el ejemplo anterior de la función **Suma**, vamos a crear una fórmula para aplicar un formato condicional a la propiedad **lblSum Color**. Vuelva a la primera pantalla de la aplicación práctica que creamos en la Unidad 3 (consulte la captura de pantalla).

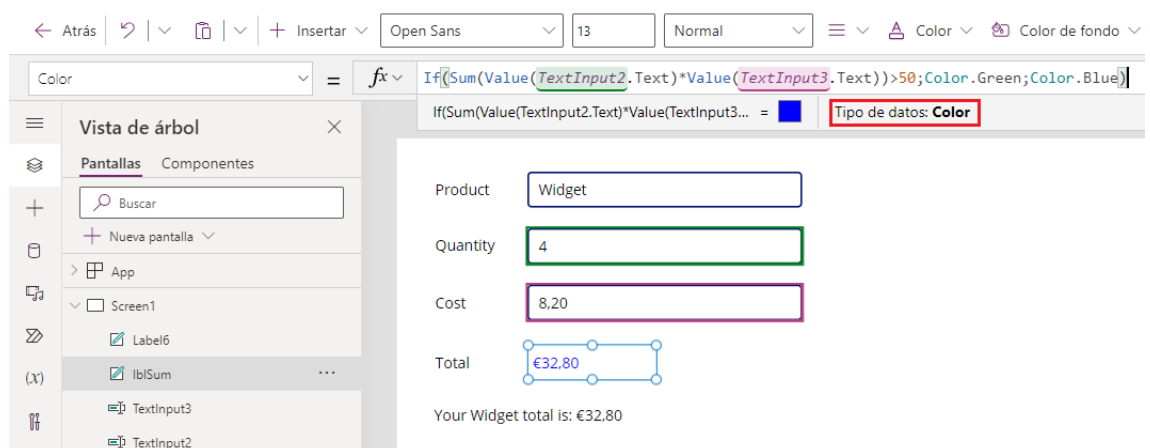
1. Seleccione **lblSum** y vaya a la propiedad **Color**. Si selecciona el campo de fórmula, observe cómo la información justo debajo del campo muestra que el valor de `RGBA(0, 0, 0, 1)` es un cuadrado de color negro y el Data type es Color.



2. Borre el valor actual y especifique lo siguiente en el campo Power *fx* para **Color**:

`If(Sum(Value(TextInput2.Text)*Value(TextInput3.Text))>50,Color.Green,Color.Blue)`

La propiedad **Color** define el color del texto en el control de etiqueta. En este ejemplo, si el cálculo **Suma** de los dos controles **TextInput** es mayor que 50, el color es verde; de lo contrario, el color es azul. Además, observe cómo, si coloca el cursor en la barra de fórmulas después de cambiar la fórmula, el Data type de salida es **Color**.



Cuando se define la propiedad **Color** en Power Apps, se puede elegir entre diferentes opciones disponibles. Puede proporcionar un objeto de color en la fórmula, como se muestra en el ejemplo, utilizando **Color.Green** o **Color.Blue** (o cualquier otro de los colores predefinidos en Power Apps).

También hay disponible una función **ColorValue** (que usa nombres de color CSS y valores de color hexadecimales de 6 y 8 dígitos) y una función **RGBA** para utilizar cualquier combinación de colores creable.

Referencia a la propiedad text para actualizar el color

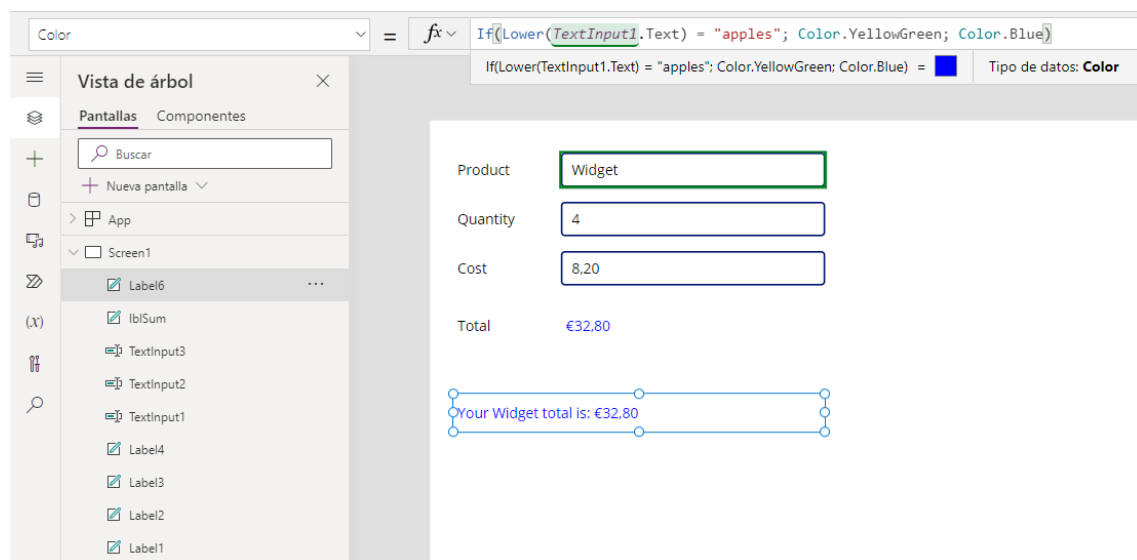
También puede hacer referencia a la propiedad **Text** para determinar la propiedad **Color**.

Imaginemos que queremos que nuestros clientes reciban comentarios en función del artículo que pidan. Si se introduce un producto concreto, cambiamos el color del texto del resumen.

Cuando se utiliza texto como valor en una fórmula, es importante distinguir entre mayúsculas y minúsculas. Por lo tanto, podemos usar una función como **Lower** para nuestra fórmula, como *Lower(yourtextvalue)*, que devolvería el valor de su texto en minúsculas.

Intente reemplazar la fórmula en la propiedad **Color** para el resumen de su transacción con lo siguiente:

```
If(Lower(TextInput1.Text) = "apples", Color.YellowGreen, Color.Blue)
```



Observe cómo el texto de la etiqueta cambia inmediatamente a azul. Ahora cambie su aplicación al modo de vista preliminar e indique "Apples" en el campo de entrada **Producto** para ver cómo el color del texto del resumen de su transacción cambia a amarillo-verde.

También observe cómo hemos introducido "apples" con una "A" mayúscula, pero la fórmula aún funcionó. La mayúscula o minúscula de nuestra entrada no importa, sólo la ortografía.

| | |
|----------|-------------------------------------|
| Product | <input type="text" value="Apples"/> |
| Quantity | <input type="text" value="4"/> |
| Cost | <input type="text" value="8,20"/> |
| Total | €32,80 |

Your Apples total is: €32,80

Probablemente, estará empezando a ver cómo el formato condicional en su aplicación de lienzo puede mejorar la experiencia del usuario. Mantenga su aplicación de práctica abierta; la volveremos a usar cuando avancemos más sobre las funciones de **Validación de datos**.

Ejemplo de IsMatch

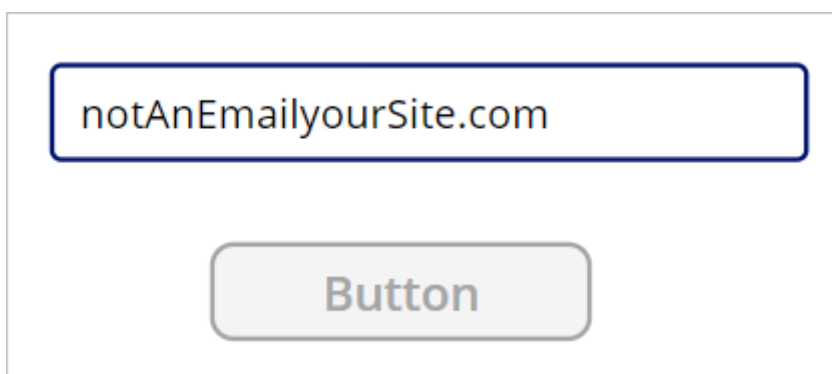
Echemos un vistazo a un ejemplo más sobre la función **IsMatch**. Puede probar esta funcionalidad para *deshabilitar* un botón si el control **Entrada de texto** no tiene especificada una dirección de correo electrónico válida. *Correo electrónico* es un patrón predefinido que se puede usar fácilmente. Sin embargo, recuerde que solo está buscando un *patrón*; no se valida si la dirección de correo electrónico realmente funciona o no. Así que vamos a utilizar la aplicación de demostración para lo siguiente:

1. Agregue un control **Text Input** al lienzo. Haga que la propiedad **Default** quede en blanco eliminando "*Entrada de texto*". Cambie el nombre del control: *tiEmailEntry*.
2. Agregue un control de **botón** justo debajo del campo de entrada de texto.
3. Modifique la propiedad **DisplayMode** del control **Botón** a la siguiente fórmula:

```
If(IsMatch(tiEmailEntry.Text, Match.Email), DisplayMode.Edit,  
DisplayMode.Disabled)
```

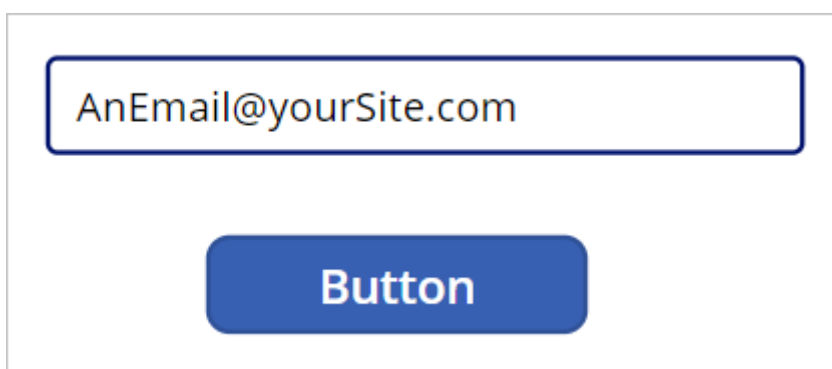
Con la fórmula de coincidencia *Email*, Power Apps prueba el valor de texto del campo de entrada con este patrón: una cadena alfanumérica, seguida del símbolo @, otra cadena alfanumérica, un punto (.) y se cierra con una cadena de varias letras. Si la cadena de entrada no coincide con el patrón especificado, el botón permanece en modo "deshabilitado" y el usuario no puede interactuar con él. Probemos la funcionalidad.

4. Obtenga una vista preliminar de la aplicación seleccionando el botón **Reproducir** situado en la esquina superior derecha.
5. Si el campo está en blanco o tiene una dirección de correo electrónico incompleta, el botón está deshabilitado y aparece en gris. Pruebe algunos ejemplos diferentes sin @ o sin el punto (.).



A screenshot of a Power Apps form. At the top, there is a text input field with a blue border containing the text "notAnEmailyourSite.com". Below the input field is a button with a light gray background and rounded corners, labeled "Button". The button is disabled.

Si escribe una dirección de correo electrónico con formato correcto, el botón cambia a azul y se puede seleccionar.



A screenshot of a Power Apps form. At the top, there is a text input field with a blue border containing the text "AnEmail@yourSite.com". Below the input field is a button with a solid blue background and rounded corners, labeled "Button". The button is enabled.

Las funciones de validación de datos son herramientas útiles para dar forma a los datos introducidos por sus usuarios. Emplearlos puede ayudar a eliminar errores de introducción de datos y mejorar aún más la experiencia del usuario.