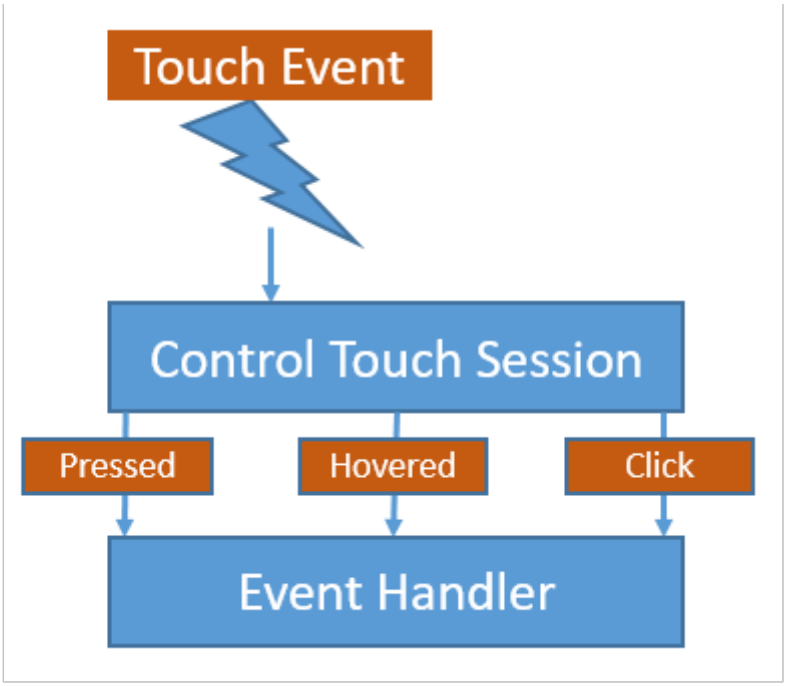
Touch Input Handling

Description

This chapter gives a detailed, technical overview of how the user touch input is handled internally which is important to understand further concepts like the Control States. The base part therefore is the CgiStudioControl::ControlTouchSession which handles the TouchEvent and emits several Events received by Controls with a Handle Control State Behavior (e.g. Control States).



Touch Event

The Touch Event is emitted by Courier and contains TouchInfo with following states:

- Move
- Down
- Up
- Hover

> Touch input can be tested with the Player which reacts to mouse input and emits Touch Events.

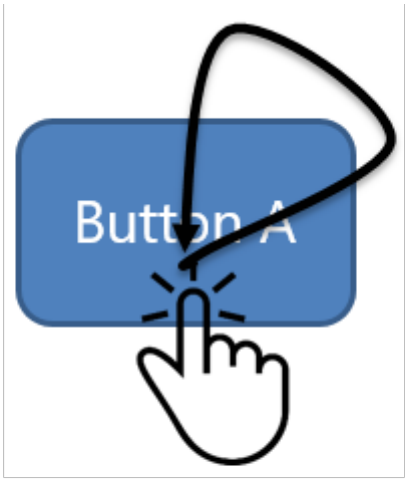
ControlTouchSession

To simplify the process especially for Control developers, the states of a TouchInfo are mapped to several, more specific Events. Therefore the ControlTouchSession generates kind of virtual input events in the sense that a combination of touch events result in a new input event (e.g. Touch Down and Touch Up on the same Control leads to a Click Event). To be able to generate these events and to send them to the right Control the ControlTouchSession contains a list to register all Control State Behaviors. So be sure that each Control which should be handled by the Control Touch Session has to contain a Control State Behavior. For each of the following events it is important to know that the events are generated by the ControlTouchSession but they are dispatched from the Control which gets touched. To understand how Events get sent also check chapter Behaviors and Events .

> Controls must have a Handle Control State Behavior to receive Events from the Control Touch Session even if the Event is not sent locally to this Behavior (e.g. the Click Event).

Pressed Event

Basically the Pressed Event which has two different states is triggered by the users touch down (state: Pressed) or by the users touch up (state: Released).
Furthermore the TouchSession reacts to the move of the touch. So if the touch moves and the primarily touched Control is not touched any longer, the ControlTouchSession emits a Pressed Event with the state Released.
If the touch is still down and enters the primarily touched Control, the TouchSession again emits a Pressed Event with the state Pressed.



A user presses Button A and drags outside and enters the area of Button A again then releases it.
So the Pressed Event will be emitted four times with following states: Pressed, Released, Pressed Released.

In many cases, especially concerning Controls, the Pressed Event gets received by the Handle Control State Behavior (see chapter Control States).

The Handle Control State Behavior even receives the Event via Local Dispatch Strategy (see Behaviors and Events). But the Event also gets dispatched via BubbleStrategy from the Control which enables other Behaviors of the Control to receive the event (e.g. PressedCondition).

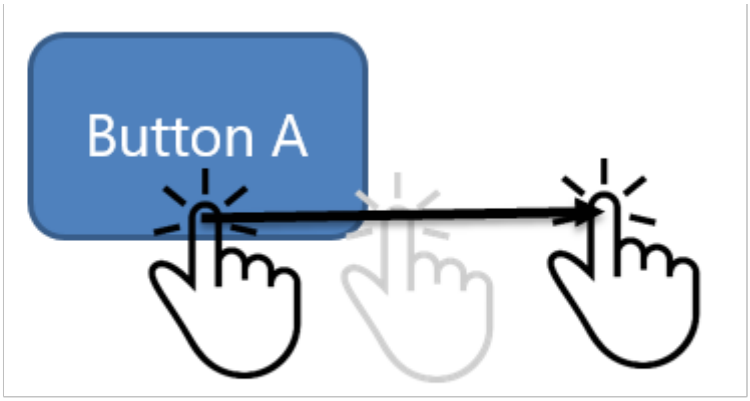
Hovered Event

Emitting the Hovered Event works similar to the Pressed Event. But in difference to the Pressed Event the Hovered Event has an additional state in case of keep hovering over the same object.

Additionally the Event is emitted when the Hover leaves the currently hovered object (**HoverLeave**) and when the Hover enters and object (**HoverEnter**).
As for the Pressed Event also the Hover Event basically gets received by the Handle Control State Behavior (see chapter Control States). But the Event can also be received by another Behavior like the HoveredCondition.

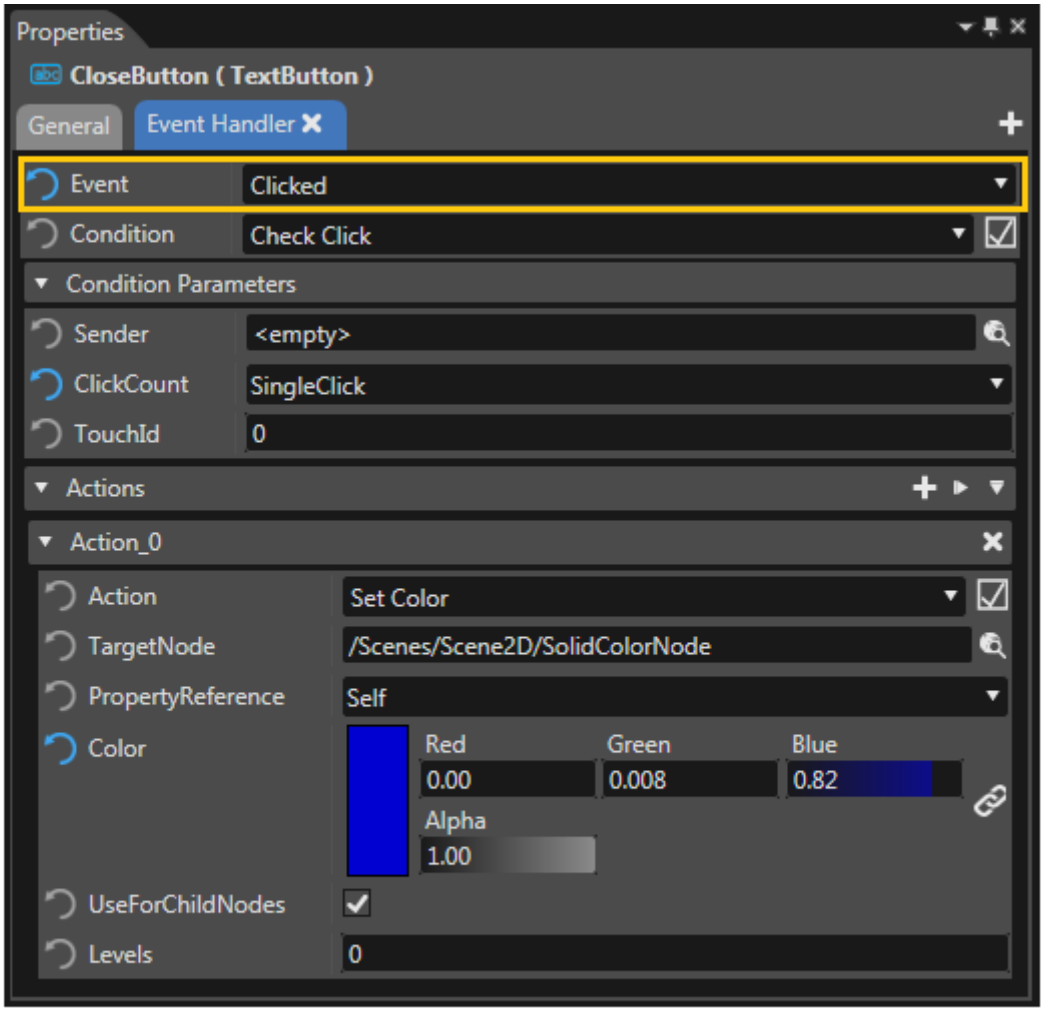
Click Event

As the previously described Events also the Click Event is generated by the ControlTouchSession but it is dispatched from the Control which gets clicked.
But different to the PressedEvent and the HoveredEvent, the ClickEvent is not sent to the Handle Control State Behavior via Local Dispatch Strategy as it is not relevant for the states of a Control. But again the Event gets dispatched from the Handle Control State Behavior which enables the user to directly react on the Event via an EventHandler (also see EventHandler Tab). Furthermore it is important to know that the Click Event is emitted only on release of the touch point. Moreover the the Control of the touch down must be the same as the Control of the touch up. So following example would not trigger a Click Event:



A user presses Button A, drags outside and then releases it.

The Click Event is often used in combination with a EventHandler Tab as shown below:



EventHandler Tab which triggers a Set Color Behavior when receiving a ClickEvent.

> Be sure to read chapter Control States and Control Examples to see how the described Events can be used.

Revision #6
Created 2023-02-16 06:59:44 UTC by Tsuyoshi.Kato
Updated 2024-07-31 23:17:13 UTC by Tsuyoshi.Kato