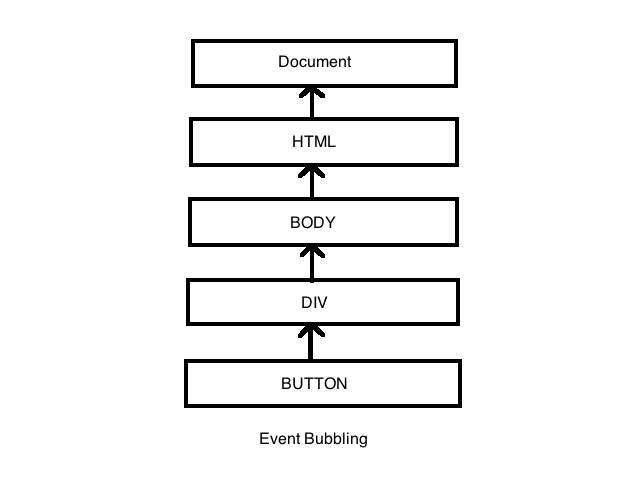
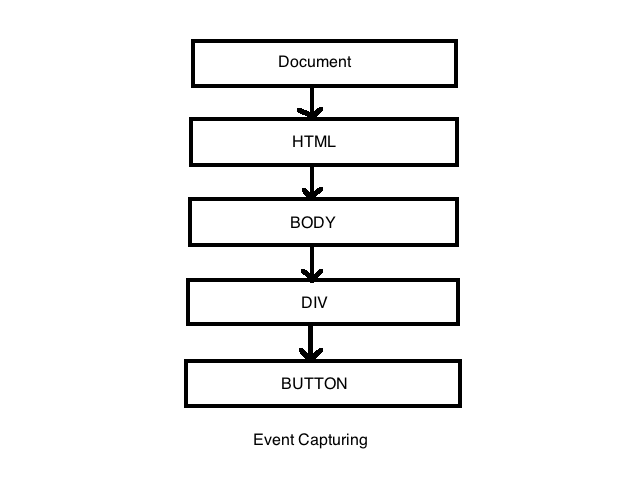
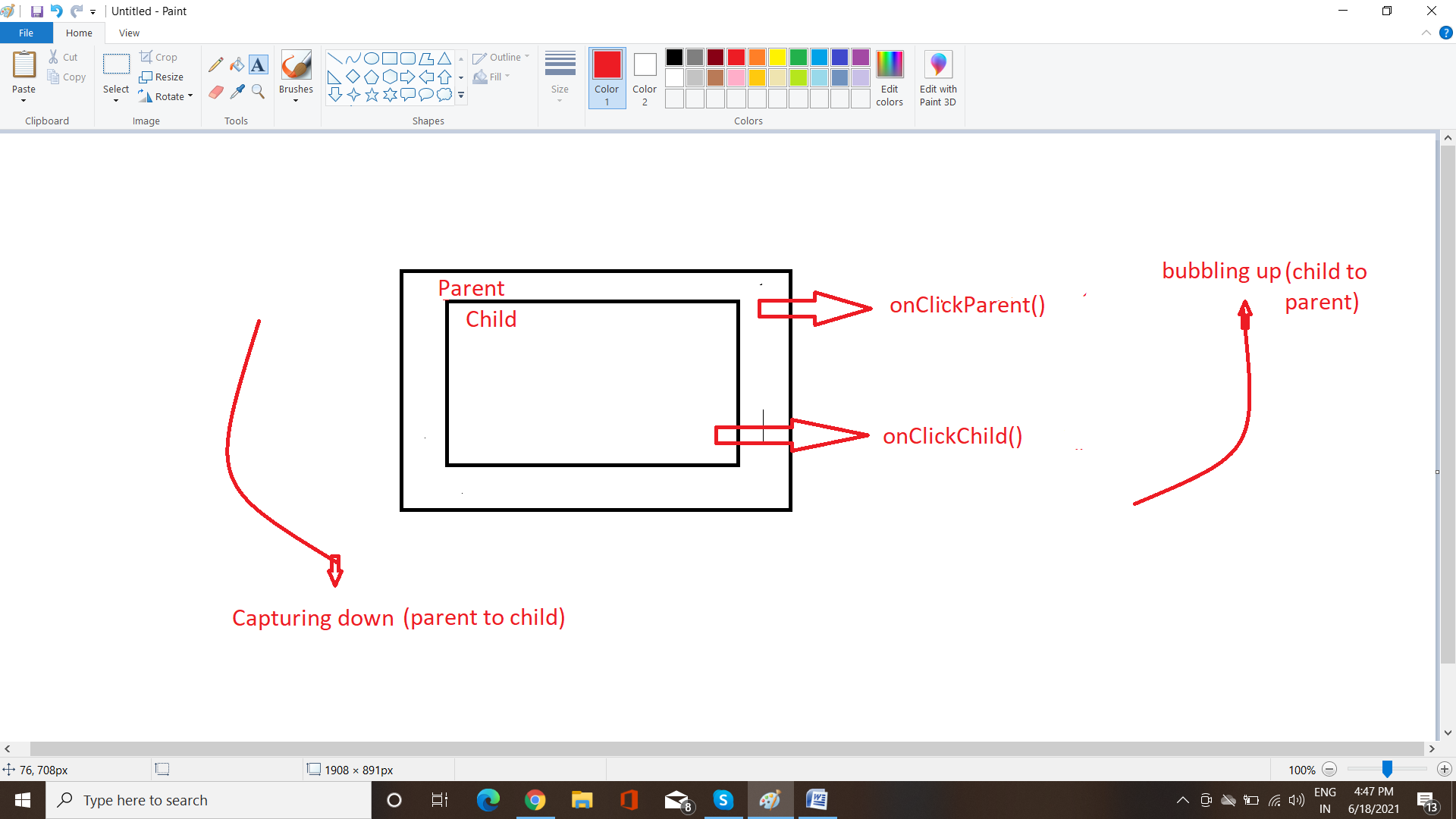
**Event Bubbling:** when an event is another element and that element is inside the other element. All the event handlers will be triggered at the same time and in event bubbling, first the child method will be called and then the parent method will be called.



**Event Capturing:** it is the opposite of event bubbling. in this, all event handlers will be triggered at the same time and in event capturing, first the parent method will be called and then child method will be called.





Like in the above diagram, the event is first captured and handled by the innermost element and then propagated to outer elements.and Event capturing is opposite to event bubbling.

Syntax : addEventListener(type, listener, useCapture)

* **type**: Use to refer to the type of event.
* **listener**: Function we want to call when the event of the specified type occurs.
* **userCapture**: Boolean value. Boolean value indicates event phase. By Default useCapture is false. It means it is in the bubbling phase.

**In program it can be written as**: addEventListener("click", function () {

            alert("You clicked the Child element!");

        }, false);

By using EventListener we can control whether to capture the events or to bubble these events.

**How to stop Event Bubbling:** to **stop** the **event bubbling**, this can be

achieved by the use of the **event**. **stopPropagation**() method. If we want to **stop** the **event** flow from **event** target to top element in DOM, **event**. **stopPropagation**() method stops the **event** to travel to the bottom to top.

**bind() :** The bind() method attaches one or more event handlers for selected elements, and specifies a function to run when the event occurs. The bind() method was deprecated in version 3.0.

syntax: $(“p”).bind(event,data,function,map)

$("p").bind("click", function(){

alert("The paragraph was clicked.");

});

**apply() :** by using apply() method in JS, we can call another object. It works while using an array{takes argument in an array}.

**Call() :** We can also use call() {take argument separately} method instead of apply().

Syntax: .apply(obj, [arr1, arr2]);

.call(obj,arr1,arr2);