

Event Handling

Events

- **events make the Flash world go ‘round**
- **based on the DOM event handling standard**
- **to make your component dispatch events and to have the abilities to add and remove event listeners, you need to implement the *IEventDispatcher* interface**
- **you can also extend the *EventDispatcher* class**
- **all listeners receive an *Event*-object**

flash.events.IEventDispatcher

```
addEventListener(type:String, listener:Function,  
    useCapture:Boolean=false, priority:int=0,  
    useWeakReference:Boolean=false):void;
```

```
dispatchEvent(event:Event):Boolean;
```

```
hasEventListener(type:String):Boolean;
```

```
removeEventListener(type:String, listener:Function,  
    useCapture:Boolean=false):void;
```

```
willTrigger(type:String):Boolean;
```

flash.events.Event

bubbles:Boolean;

cancelable:Boolean;

currentTarget:Boolean;

eventPhase:uint;

target:Object;

type:String;

**Indicates whether an event is
bubbling event**

flash.events.Event

```
bubbles:Boolean;
```

```
cancelable:Boolean;
```

```
currentTarget:Boolean;
```

```
eventPhase:uint;
```

```
target:Object;
```

```
type:String;
```

Indicates whether the behavior associated with the event can be prevented

flash.events.Event

```
bubbles:Boolean;
```

```
cancelable:Boolean;
```

```
currentTarget:Object;
```

```
eventPhase:uint;
```

```
target:Object;
```

```
type:String;
```

**The object that is actively processing
the event with an event listener**

flash.events.Event

```
bubbles:Boolean;
```

```
cancelable:Boolean;
```

```
currentTarget:Boolean;
```

```
eventPhase:uint;
```

```
target:Object;
```

```
type:String;
```

The current phase in the event flow:

- **EventPhase.CAPTURING_PHASE**
- **EventPhase.AT_TARGET**
- **EventPhase.BUBBLING_PHASE**

flash.events.Event

```
bubbles:Boolean;
```

```
cancelable:Boolean;
```

```
currentTarget:Boolean;
```

```
eventPhase:uint;
```

```
target:Object;
```

```
type:String;
```

The display list node containing the dispatcher of the event

flash.events.Event

```
bubbles:Boolean;
```

```
cancelable:Boolean;
```

```
currentTarget:Boolean;
```

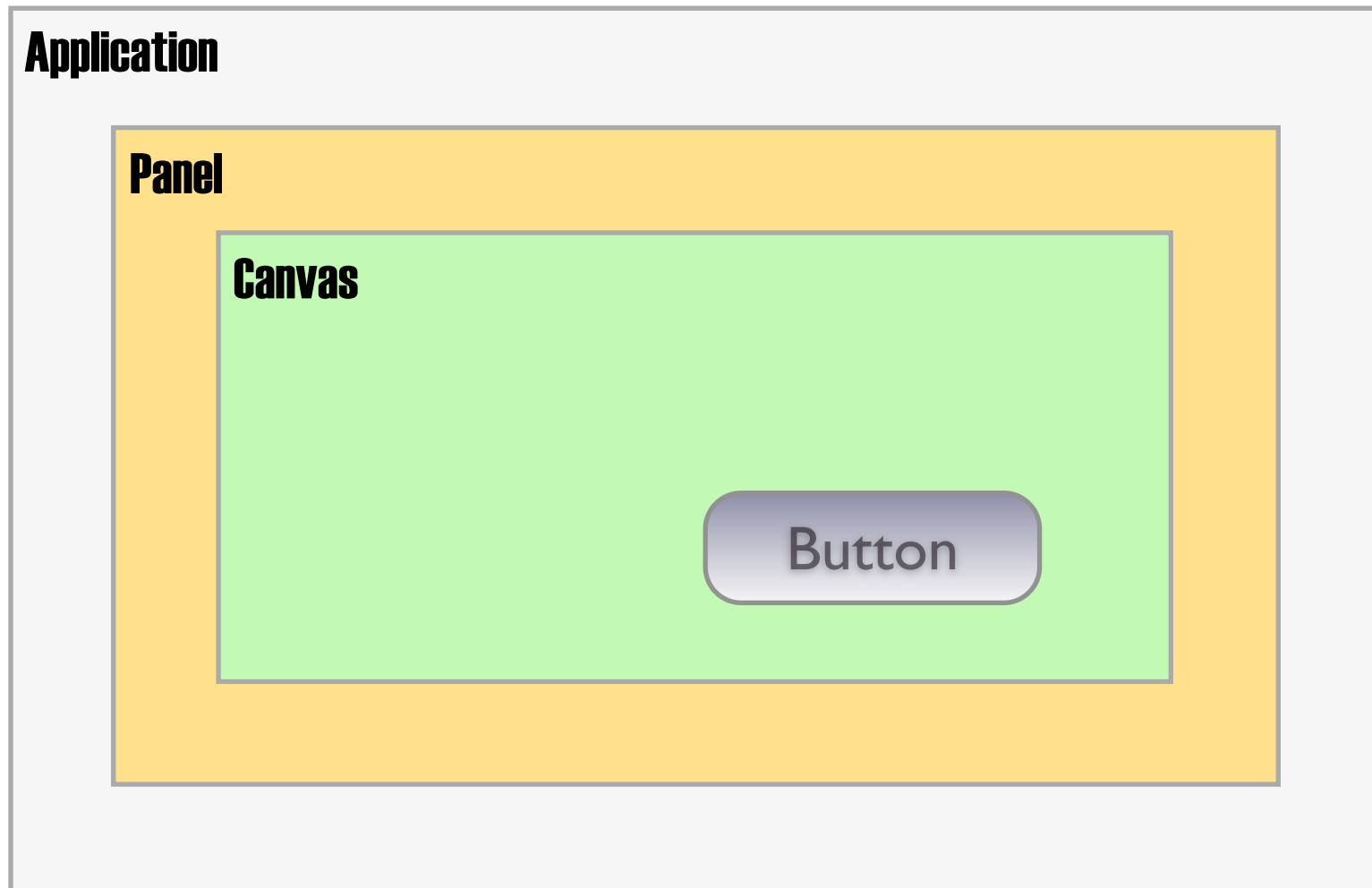
```
eventPhase:uint;
```

```
target:Object;
```

```
type:String;
```

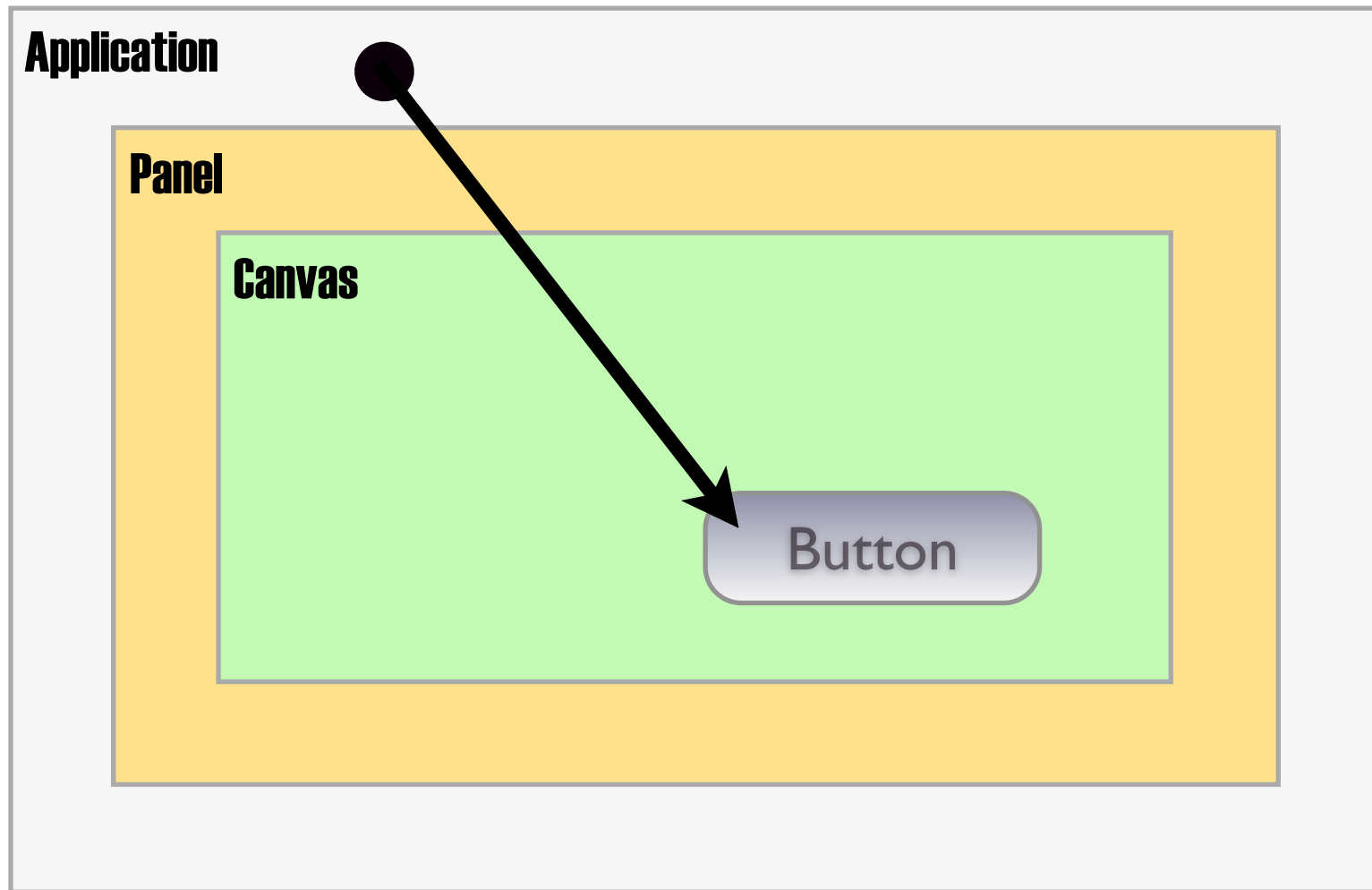
The name of the event (it is case-sensitive)

Event Propagation



- *display list* - hierarchical container of all visual objects
- when an event occurs it is not dispatched directly to the movie clip, but rather to the display list

Event Propagation

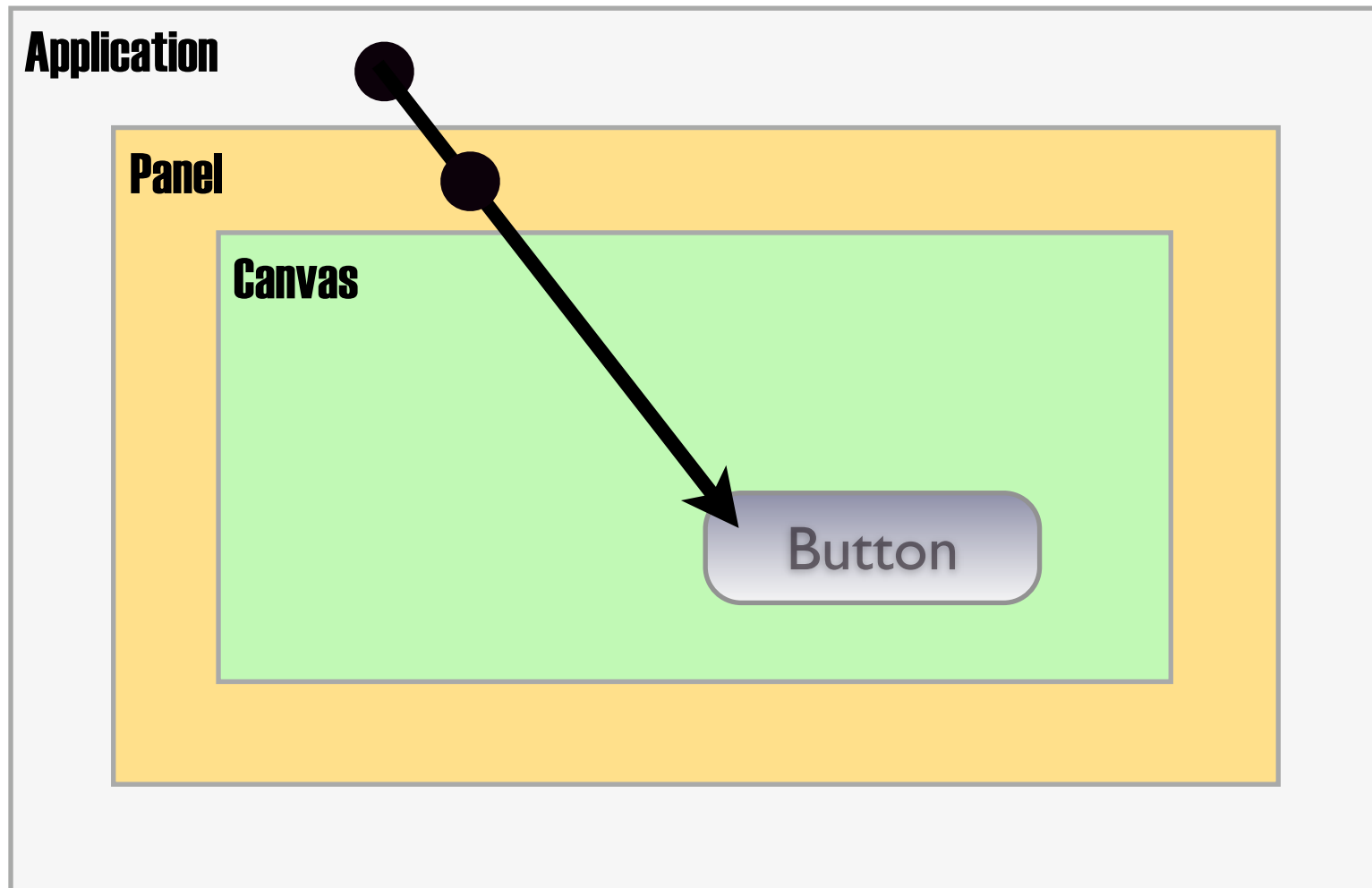


```
eventPhase = EventPhase.CAPTURING_PHASE;
```

```
target = button;
```

```
currentTarget = application;
```

Event Propagation

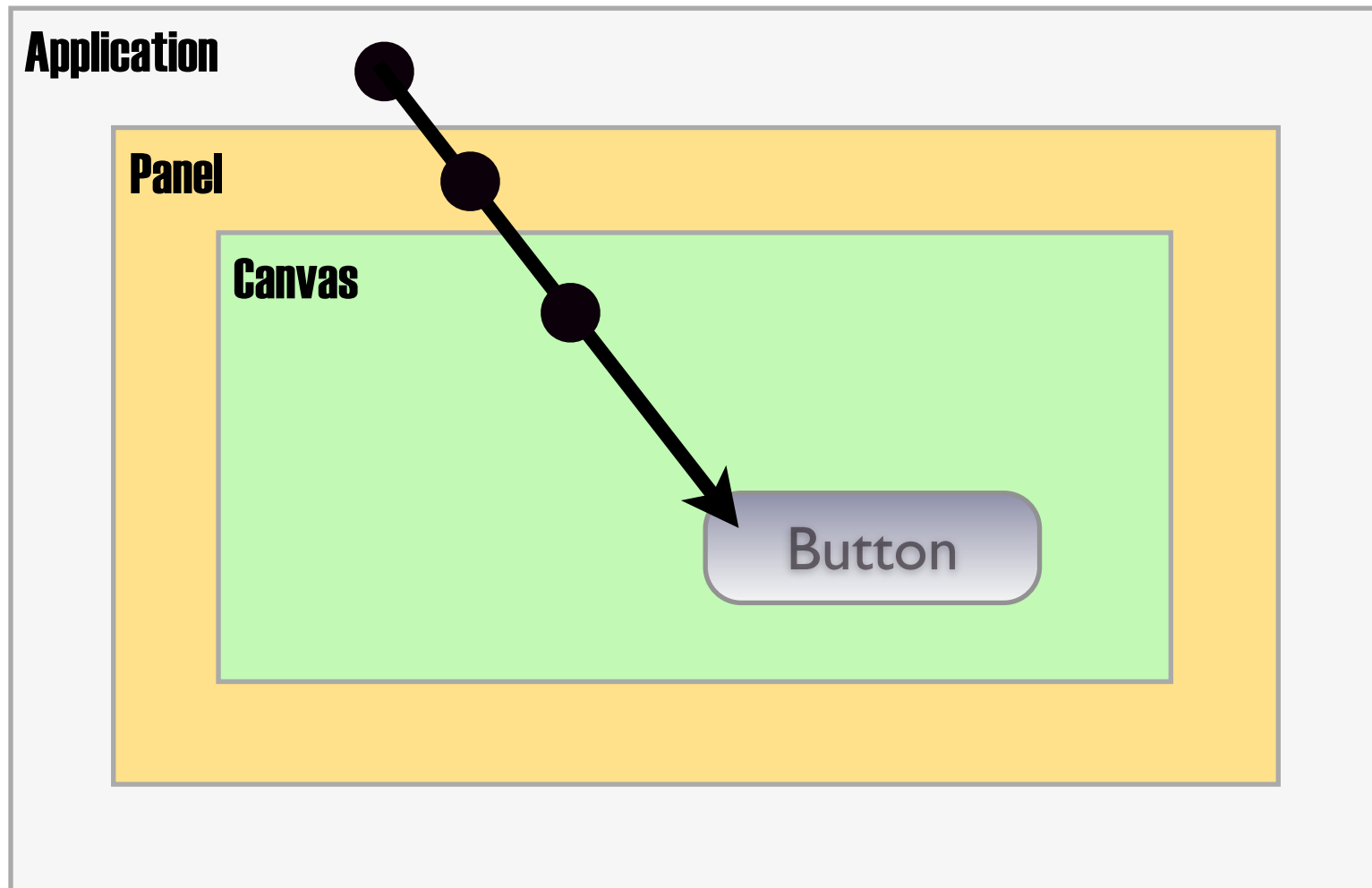


```
eventPhase = EventPhase.CAPTURING_PHASE;
```

```
target = button;
```

```
currentTarget = panel;
```

Event Propagation

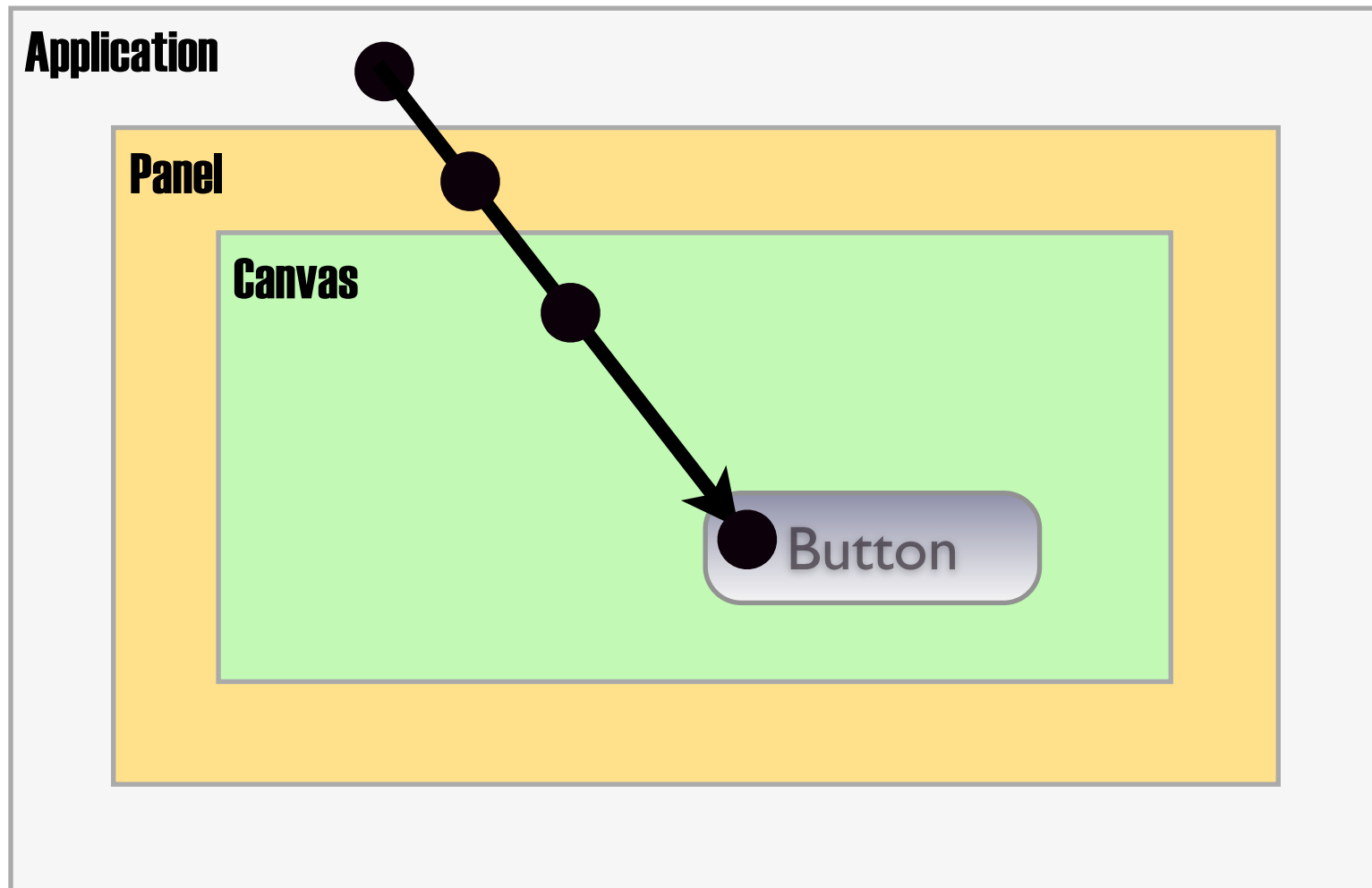


```
eventPhase = EventPhase.CAPTURING_PHASE;
```

```
target = button;
```

```
currentTarget = canvas;
```

Event Propagation

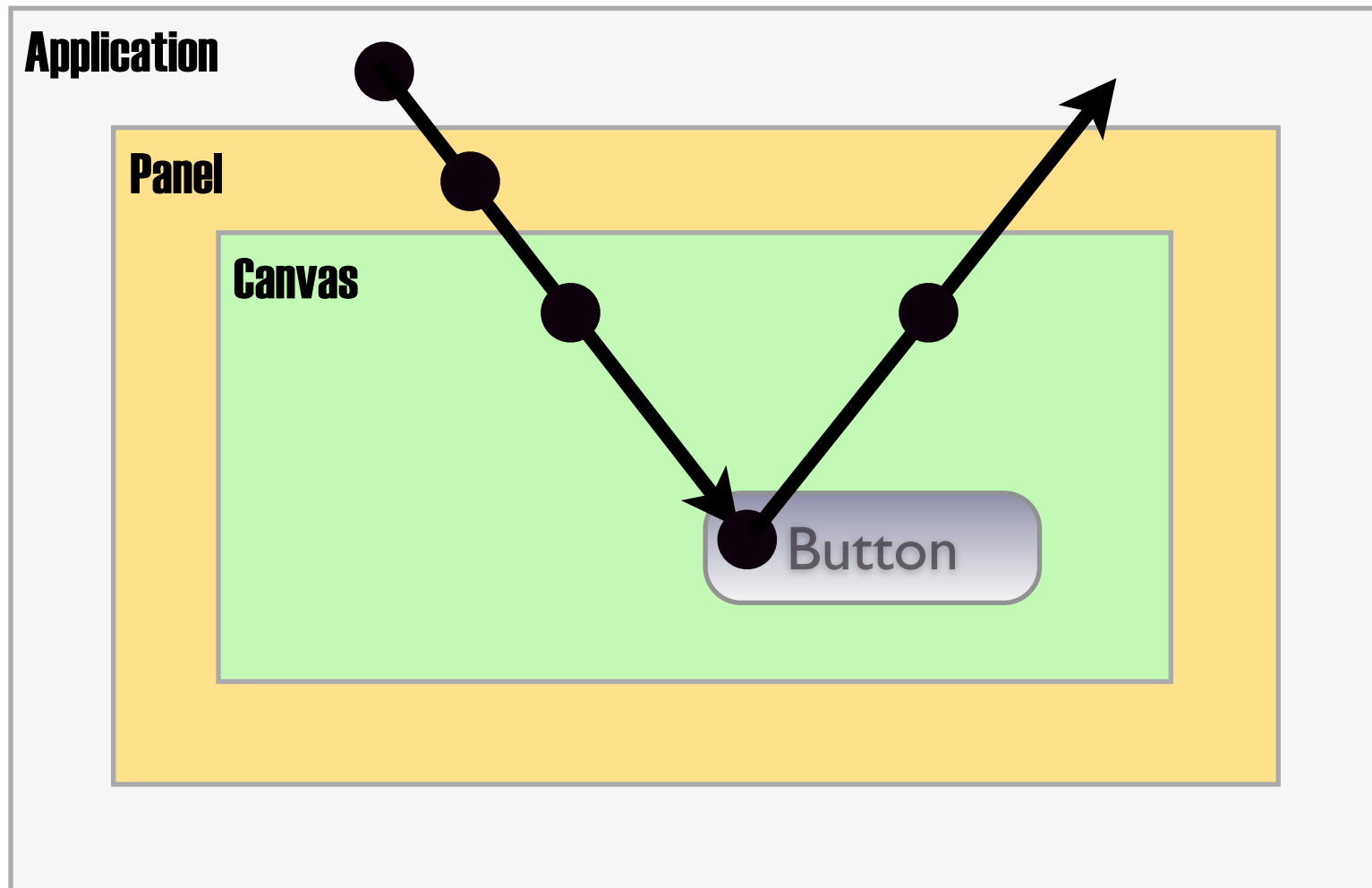


```
eventPhase = EventPhase.AT_TARGET;
```

```
target = button;
```

```
currentTarget = button;
```

Event Propagation

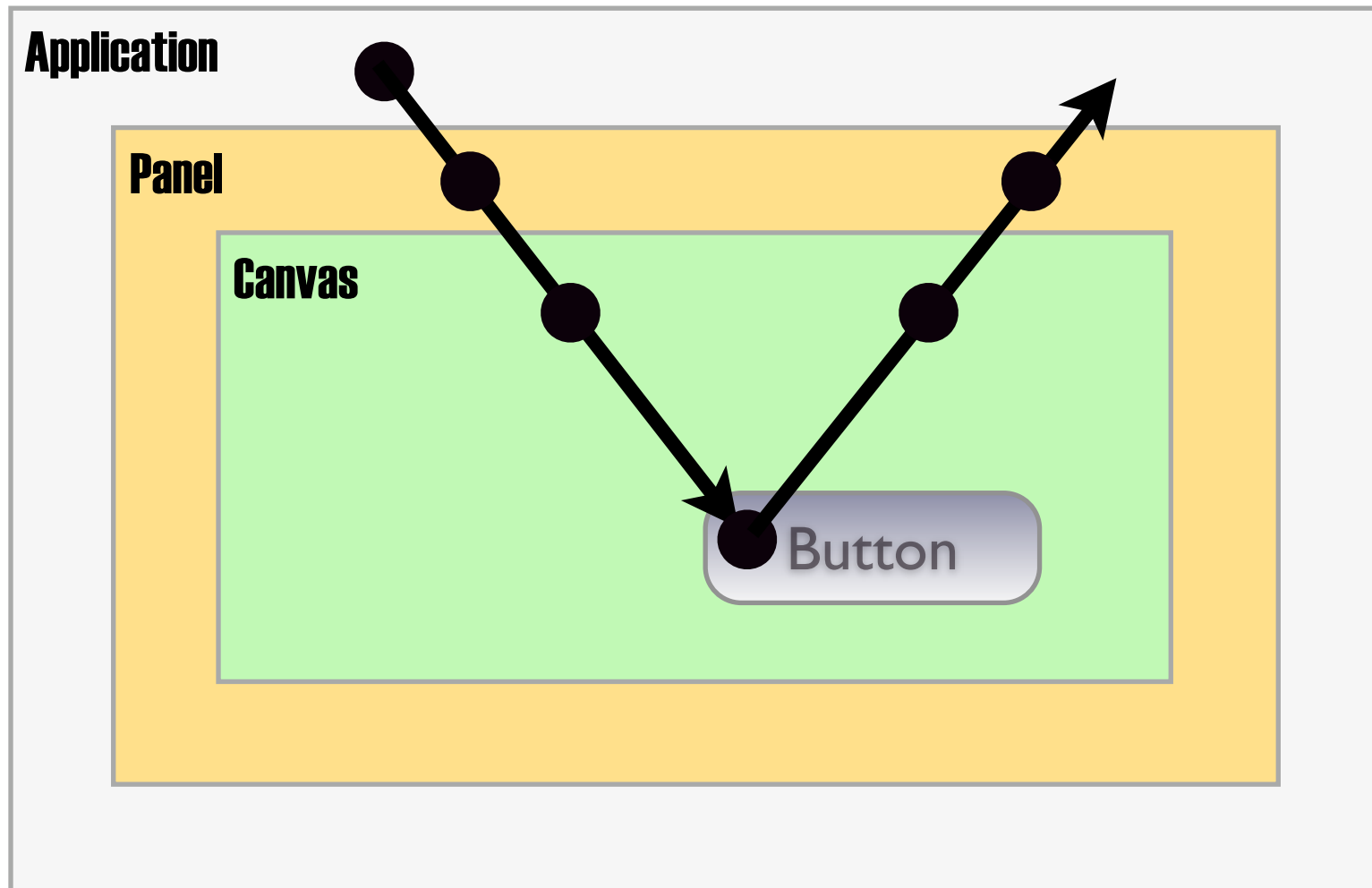


```
eventPhase = EventPhase.BUBBLING_PHASE;
```

```
target = button;
```

```
currentTarget = canvas;
```

Event Propagation

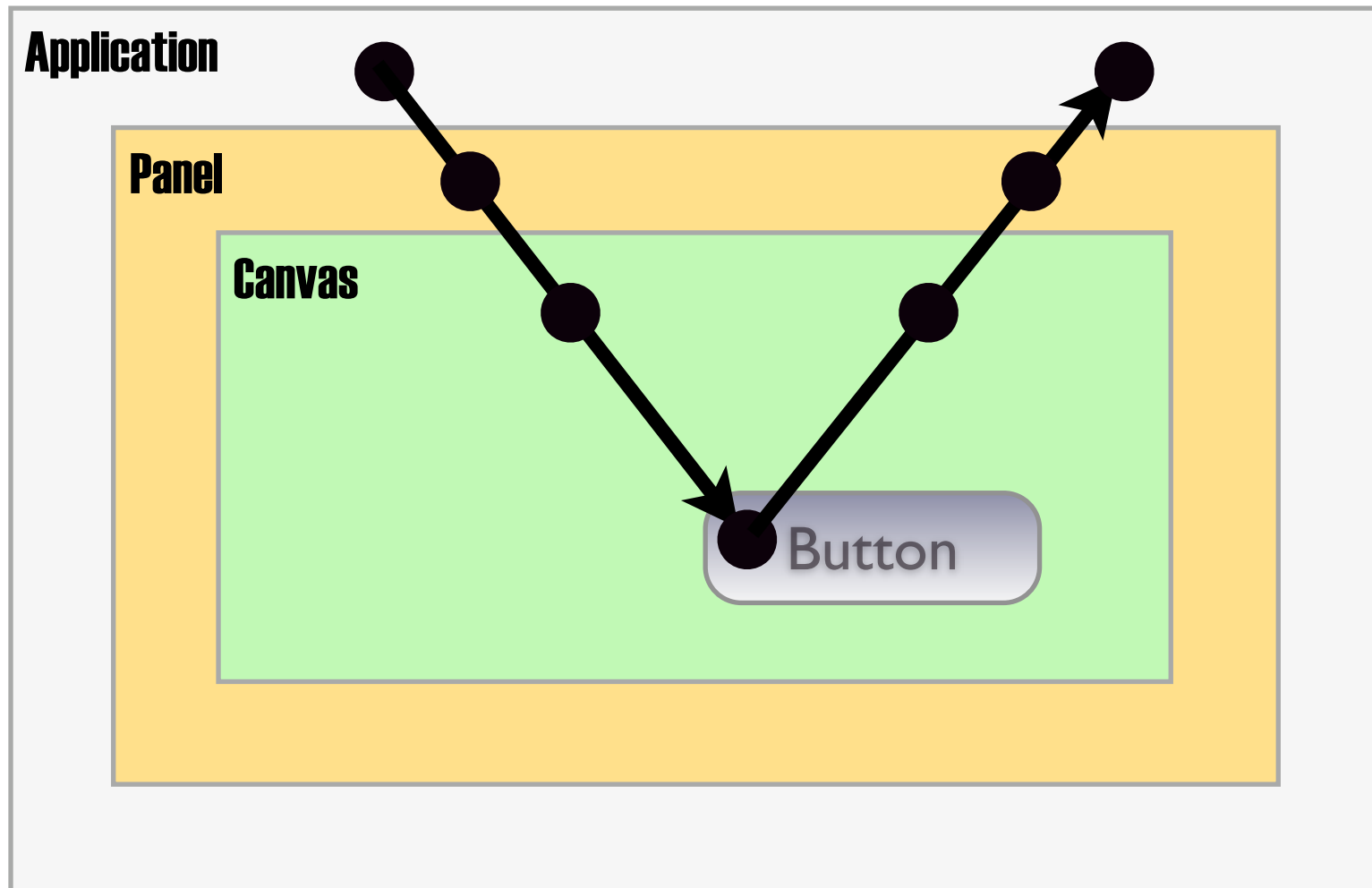


```
eventPhase = EventPhase.BUBBLING_PHASE;
```

```
target = button;
```

```
currentTarget = panel;
```

Event Propagation



```
eventPhase = EventPhase.BUBBLING_PHASE;
```

```
target = button;
```

```
currentTarget = application;
```

```
event.stopPropagation();
```

- **during any phase you can stop the traversal of the display list and prevent an event from continuing on its way through the event flow**
- **allows the current node to execute its listeners**

```
event.stopImmediatePropagation();
```

- **does not allow any other event listeners on the current node to execute**

```
event.preventDefault();
```

- **many events have associated behaviors that Flash Player carries out by default (e.g. when a user types a character in an input field that character is displayed in the input's text field)**
- **you can prevent the event's default behavior if that behavior can be canceled**

Summary

- **events overview**
- **flash.events.IEventDispatcher**
- **flash.events.Event**
- **event propagation**
- **stopping the propagation and prevent default**