IIT Bombay

Computer Programming

Dr. Deepak B Phatak
Dr. Supratik Chakraborty
Department of Computer Science and Engineering
IIT Bombay

Session: Advanced Graphics Events
Guest Lecturer: Dr. Abhiram Ranade

Quick recap

IIT Bombay

Quick recap

IIT Bombay

Last session: The `getClick` function for waiting for clicks.

Quick recap

IIT Bombay

Last session: The `getClick` function for waiting for clicks.

This session: How to wait for other kinds of events.

Quick recap

IIT Bombay

Last session: The `getClick` function for waiting for clicks.

This session: How to wait for other kinds of events.

Reference: Chapter 20 of "An introduction to programming through C++", McGraw Hill Education, 2014.

Basics of Events

IIT Bombay

The term *event* is used in Simplecpp to denote

Basics of Events

IIT Bombay

The term *event* is used in Simplecpp to denote

- Pressing of a button of the mouse.

Basics of Events

IIT Bombay

The term *event* is used in Simplecpp to denote

- Pressing of a button of the mouse.
- Release of a button of the mouse.

Basics of Events

IIT Bombay

The term *event* is used in Simplecpp to denote

- Pressing of a button of the mouse.
- Release of a button of the mouse.
- *Dragging* of the mouse.
  Dragging = moving the mouse with a button pressed.

Basics of Events

IIT Bombay

The term *event* is used in Simplecpp to denote

- Pressing of a button of the mouse.
- Release of a button of the mouse.
- *Dragging* of the mouse.
  Dragging = moving the mouse with a button pressed.
- Pressing of a key on the keyboard.

Basics of Events

IIT Bombay

The term *event* is used in Simplecpp to denote

- Pressing of a button of the mouse.
- Release of a button of the mouse.
- *Dragging* of the mouse.
  Dragging = moving the mouse with a button pressed.
- Pressing of a key on the keyboard.

A C++ program can choose to wait for above events to happen, and once they happen, can find out what has happened, and then continue.

Basics of Events

IIT Bombay

The term *event* is used in Simplecpp to denote

- Pressing of a button of the mouse.
- Release of a button of the mouse.
- *Dragging* of the mouse.
  Dragging = moving the mouse with a button pressed.
- Pressing of a key on the keyboard.

A C++ program can choose to wait for above events to happen, and once they happen, can find out what has happened, and then continue.

A C++ program can also merely *check* whether any of the above events has already happened, without any waiting.

Basics of Events

IIT Bombay

The term *event* is used in Simplecpp to denote

- Pressing of a button of the mouse.
- Release of a button of the mouse.
- *Dragging* of the mouse.
  Dragging = moving the mouse with a button pressed.
- Pressing of a key on the keyboard.

A C++ program can choose to wait for above events to happen, and once they happen, can find out what has happened, and then continue.

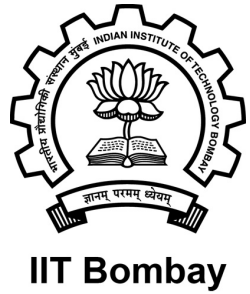
A C++ program can also merely *check* whether any of the above events has already happened, without any waiting.

If there are many windows on the screen, then you must first click on the canvas so that subsequent events will be detected by your program.

Event Objects

IIT Bombay

Event Objects



IIT Bombay

Objects of (built-in) class `XEvent` are used for holding information about events.

Event Objects

IIT Bombay

Objects of (built-in) class `XEvent` are used for holding information about events.

You do not need to know the exact definition of `XEvent`. Some functions etc. are provided which enable you to get the required information.

Waiting for an event

IIT Bombay

Waiting for an event

IIT Bombay

The function `nextEvent` is used for waiting for an event.

Waiting for an event

IIT Bombay

The function `nextEvent` is used for waiting for an event.
It takes as argument a reference to an XEvent object.

Waiting for an event

IIT Bombay

The function `nextEvent` is used for waiting for an event. It takes as argument a reference to an XEvent object.

Typical use:

```
XEvent e1;
nextEvent(e1);
```

Waiting for an event

IIT Bombay

The function `nextEvent` is used for waiting for an event.
It takes as argument a reference to an XEvent object.

Typical use:

```
XEvent e1;
nextEvent(e1);
```

The call causes the program to wait for an event to happen.

Waiting for an event

IIT Bombay

The function `nextEvent` is used for waiting for an event.
It takes as argument a reference to an XEvent object.

Typical use:

```
XEvent e1;
nextEvent(e1);
```

The call causes the program to wait for an event to happen.
Calling `nextEvent` is like waiting for the user to type in data, i.e. executing `cin >> ...`

Waiting for an event

IIT Bombay

The function `nextEvent` is used for waiting for an event.
It takes as argument a reference to an XEvent object.

Typical use:

```
XEvent e1;
nextEvent(e1);
```

The call causes the program to wait for an event to happen.
Calling `nextEvent` is like waiting for the user to type in data, i.e. executing `cin >> ...`

When some event finally happens, the information about it is put into the passed `XEvent` object, in this case `e1`.

Waiting for an event

IIT Bombay

The function `nextEvent` is used for waiting for an event.
It takes as argument a reference to an XEvent object.

Typical use:

```
XEvent e1;
nextEvent(e1);
```

The call causes the program to wait for an event to happen.
Calling `nextEvent` is like waiting for the user to type in data, i.e. executing `cin >> ...`

When some event finally happens, the information about it is put into the passed `XEvent` object, in this case `e1`.
After this execution continues.

Getting information about events

IIT Bombay

Getting information about events

IIT Bombay

The following functions are available

Getting information about events

IIT Bombay

The following functions are available

- `bool mouseButtonPressEvent(XEvent &e1);`

Getting information about events

IIT Bombay

The following functions are available

- `bool mouseButtonPressEvent(XEvent &e1);`
  Returns true iff `e1` involves some mouse button being pressed.

Getting information about events

IIT Bombay

The following functions are available

- `bool mouseButtonPressEvent(XEvent &e1);`
  Returns true iff `e1` involves some mouse button being pressed.
- `bool mouseButtonReleaseEvent(XEvent &e1);`

Getting information about events

IIT Bombay

The following functions are available

- `bool mouseButtonPressEvent(XEvent &e1);`
  Returns true iff `e1` involves some mouse button being pressed.
- `bool mouseButtonReleaseEvent(XEvent &e1);`
  Returns true iff `e1` involves some mouse button being released.

Getting information about events

IIT Bombay

The following functions are available

- `bool mouseButtonPressEvent(XEvent &e1);`
  Returns true iff `e1` involves some mouse button being pressed.
- `bool mouseButtonReleaseEvent(XEvent &e1);`
  Returns true iff `e1` involves some mouse button being released.
- `bool mouseDragEvent(XEvent &e1);`

Getting information about events

IIT Bombay

The following functions are available

- `bool mouseButtonPressEvent(XEvent &e1);`
  Returns true iff `e1` involves some mouse button being pressed.
- `bool mouseButtonReleaseEvent(XEvent &e1);`
  Returns true iff `e1` involves some mouse button being released.
- `bool mouseDragEvent(XEvent &e1);`
  Returns true iff `e1` involves the mouse being dragged.

Getting information about events

IIT Bombay

The following functions are available

- `bool mouseButtonPressEvent(XEvent &e1);`
  Returns true iff `e1` involves some mouse button being pressed.
- `bool mouseButtonReleaseEvent(XEvent &e1);`
  Returns true iff `e1` involves some mouse button being released.
- `bool mouseDragEvent(XEvent &e1);`
  Returns true iff `e1` involves the mouse being dragged.
- `bool keyPressEvent(XEvent &e1);`

Getting information about events

IIT Bombay

The following functions are available

- `bool mouseButtonPressEvent(XEvent &e1);`
  Returns true iff `e1` involves some mouse button being pressed.
- `bool mouseButtonReleaseEvent(XEvent &e1);`
  Returns true iff `e1` involves some mouse button being released.
- `bool mouseDragEvent(XEvent &e1);`
  Returns true iff `e1` involves the mouse being dragged.
- `bool keyPressEvent(XEvent &e1);`
  Returns true iff `e1` involves some key being pressed.

Getting information about events

IIT Bombay

The following functions are available

- `bool mouseButtonPressEvent(XEvent &e1);`
  Returns true iff `e1` involves some mouse button being pressed.
- `bool mouseButtonReleaseEvent(XEvent &e1);`
  Returns true iff `e1` involves some mouse button being released.
- `bool mouseDragEvent(XEvent &e1);`
  Returns true iff `e1` involves the mouse being dragged.
- `bool keyPressEvent(XEvent &e1);`
  Returns true iff `e1` involves some key being pressed.

How to get more detailed information: next.

Getting more information about an event e1

IIT Bombay

Getting more information about an event e1

IIT Bombay

The following data members can be accessed.

Getting more information about an event e1

IIT Bombay

The following data members can be accessed.

- `e1.xbutton.button` : Equals 1, 2, 3 depending upon which button was pressed or released, assuming event `e1` involved pressing or releasing a mouse button.

Getting more information about an event e1

IIT Bombay

The following data members can be accessed.

- `e1.xbutton.button` : Equals 1, 2, 3 depending upon which button was pressed or released, assuming event `e1` involved pressing or releasing a mouse button.
- `e1.xbutton.x` : Equals the x-coordinate of the position where `e1` happened.

Getting more information about an event e1

IIT Bombay

The following data members can be accessed.

- `e1.xbutton.button` : Equals 1, 2, 3 depending upon which button was pressed or released, assuming event `e1` involved pressing or releasing a mouse button.
- `e1.xbutton.x` : Equals the x-coordinate of the position where `e1` happened.
- `e1.xbutton.y` : Equals the y-coordinate of the position where `e1` happened.

Getting more information about an event e1

IIT Bombay

The following data members can be accessed.

- `e1.xbutton.button` : Equals 1, 2, 3 depending upon which button was pressed or released, assuming event `e1` involved pressing or releasing a mouse button.
- `e1.xbutton.x` : Equals the x-coordinate of the position where `e1` happened.
- `e1.xbutton.y` : Equals the y-coordinate of the position where `e1` happened.
- `e1.xmotion.x` : Equals the x-coordinate of where dragging happened, in case `e1` was a mouse drag event.

Getting more information about an event e1

IIT Bombay

The following data members can be accessed.

- `e1.xbutton.button` : Equals 1, 2, 3 depending upon which button was pressed or released, assuming event `e1` involved pressing or releasing a mouse button.
- `e1.xbutton.x` : Equals the x-coordinate of the position where `e1` happened.
- `e1.xbutton.y` : Equals the y-coordinate of the position where `e1` happened.
- `e1.xmotion.x` : Equals the x-coordinate of where dragging happened, in case `e1` was a mouse drag event.
- `e1.xmotion.y` : Equals the y-coordinate of where dragging happened, in case `e1` was a mouse drag event.

Getting more information about an event e1

IIT Bombay

Getting more information about an event e1

IIT Bombay

The function

Getting more information about an event e1

IIT Bombay

The function

```
char charFromEvent(XEvent &e1);
```

Getting more information about an event e1

IIT Bombay

The function

```
char charFromEvent(XEvent &e1);
```

returns the ASCII value of the key pressed, in case e1 is a key press event.

Getting more information about an event e1

IIT Bombay

The function

```
char charFromEvent(XEvent &e1);
```

returns the ASCII value of the key pressed, in case e1 is a key press event.
The following data members are also useful.

Getting more information about an event e1

IIT Bombay

The function

```
char charFromEvent(XEvent &e1);
```

returns the ASCII value of the key pressed, in case e1 is a key press event.
The following data members are also useful.

- `e1.xkey.x` : Equals the x-coordinate of the cursor position when e1 happened.

Getting more information about an event e1

IIT Bombay

The function

```
char charFromEvent(XEvent &e1);
```

returns the ASCII value of the key pressed, in case e1 is a key press event.
The following data members are also useful.

- `e1.xkey.x` : Equals the x-coordinate of the cursor position when e1 happened.
- `e1.xkey.y` : Equals the y-coordinate of the cursor position when e1 happened.

Getting more information about an event e1

IIT Bombay

The function

```
char charFromEvent(XEvent &e1);
```

returns the ASCII value of the key pressed, in case e1 is a key press event.
The following data members are also useful.

- `e1.xkey.x` : Equals the x-coordinate of the cursor position when e1 happened.
- `e1.xkey.y` : Equals the y-coordinate of the cursor position when e1 happened.

Key press events may not be detected properly if "caps lock" or "Num lock" modes are on. Remove these first.

Checking for events

IIT Bombay

Checking for events

IIT Bombay

Function `checkEvent` checks if an event has happened.

Checking for events

IIT Bombay

Function `checkEvent` checks if an event has happened.
It takes as argument a reference to an `XEvent` object.

Checking for events

IIT Bombay

Function `checkEvent` checks if an event has happened.
It takes as argument a reference to an `XEvent` object.
It returns `true` iff an event has happened after the last call to `nextEvent` and has not been yet reported in any `checkEvent`.

Checking for events

IIT Bombay

Function `checkEvent` checks if an event has happened.
It takes as argument a reference to an `XEvent` object.
It returns `true` iff an event has happened after the last call to `nextEvent` and has not been yet reported in any `checkEvent`.

Typical use:

```
XEvent e1;
if(checkEvent(e1)){ ..A..}
else { ..B..}
```

Checking for events

IIT Bombay

Function `checkEvent` checks if an event has happened.
It takes as argument a reference to an `XEvent` object.
It returns `true` iff an event has happened after the last call to `nextEvent` and has not been yet reported in any `checkEvent`.

Typical use:

```
XEvent e1;
if(checkEvent(e1)){ ..A..}
else { ..B..}
```

The call to `checkEvent` does not wait; either code `A` or code `B` is immediately executed, depending upon whether the event happened.

Checking for events

IIT Bombay

Function `checkEvent` checks if an event has happened.
It takes as argument a reference to an `XEvent` object.
It returns `true` iff an event has happened after the last call to `nextEvent` and has not been yet reported in any `checkEvent`.

Typical use:

```
XEvent e1;
if(checkEvent(e1)){ ..A..}
else { ..B..}
```

The call to `checkEvent` does not wait; either code `A` or code `B` is immediately executed, depending upon whether the event happened.

Information about the event that happened (if any) can be obtained using the functions and members described earlier.

An example program

IIT Bombay

An example program

IIT Bombay

We will write a program that allows you to draw on the canvas.

An example program

IIT Bombay

We will write a program that allows you to draw on the canvas.

- Drawing starts when you press a mouse button.

An example program

IIT Bombay

We will write a program that allows you to draw on the canvas.

- Drawing starts when you press a mouse button.
- If you drag the mouse, then the line follows the path taken by the mouse.

An example program

IIT Bombay

We will write a program that allows you to draw on the canvas.

- Drawing starts when you press a mouse button.
- If you drag the mouse, then the line follows the path taken by the mouse.
- The drawing stops when the mouse button is released.

An example program

IIT Bombay

We will write a program that allows you to draw on the canvas.

- Drawing starts when you press a mouse button.
- If you drag the mouse, then the line follows the path taken by the mouse.
- The drawing stops when the mouse button is released.
- If you merely move the mouse without pressing any button, no line is drawn.

An example program

IIT Bombay

We will write a program that allows you to draw on the canvas.

- Drawing starts when you press a mouse button.
- If you drag the mouse, then the line follows the path taken by the mouse.
- The drawing stops when the mouse button is released.
- If you merely move the mouse without pressing any button, no line is drawn.
- If the escape key is pressed, the program ends.

Summary

IIT Bombay

We discussed how to handle mouse and keyboard events.

Summary

IIT Bombay

We discussed how to handle mouse and keyboard events.

- `nextEvent` enables waiting for mouse and keyboard events.

Summary

IIT Bombay

We discussed how to handle mouse and keyboard events.

- `nextEvent` enables waiting for mouse and keyboard events.
- `checkEvent` enables determining if an event has already happened.

Summary

IIT Bombay

We discussed how to handle mouse and keyboard events.

- `nextEvent` enables waiting for mouse and keyboard events.
- `checkEvent` enables determining if an event has already happened.
- Information about events can be obtained by calling functions on the event object, or examining its members.

Summary

IIT Bombay

We discussed how to handle mouse and keyboard events.

- `nextEvent` enables waiting for mouse and keyboard events.
- `checkEvent` enables determining if an event has already happened.
- Information about events can be obtained by calling functions on the event object, or examining its members.
- Chapter 20 gives an example of a "Snake" game that can be developed. Other games are also possible.

Summary

IIT Bombay

We discussed how to handle mouse and keyboard events.

- `nextEvent` enables waiting for mouse and keyboard events.
- `checkEvent` enables determining if an event has already happened.
- Information about events can be obtained by calling functions on the event object, or examining its members.
- Chapter 20 gives an example of a "Snake" game that can be developed. Other games are also possible.
- The drawing program developed above can be extended to recognize what is drawn: Is the user writing something? Is the user drawing a circle?

Summary

IIT Bombay

We discussed how to handle mouse and keyboard events.

- `nextEvent` enables waiting for mouse and keyboard events.
- `checkEvent` enables determining if an event has already happened.
- Information about events can be obtained by calling functions on the event object, or examining its members.
- Chapter 20 gives an example of a "Snake" game that can be developed. Other games are also possible.
- The drawing program developed above can be extended to recognize what is drawn: Is the user writing something? Is the user drawing a circle?
- The possibilities for doing interesting projects are endless..