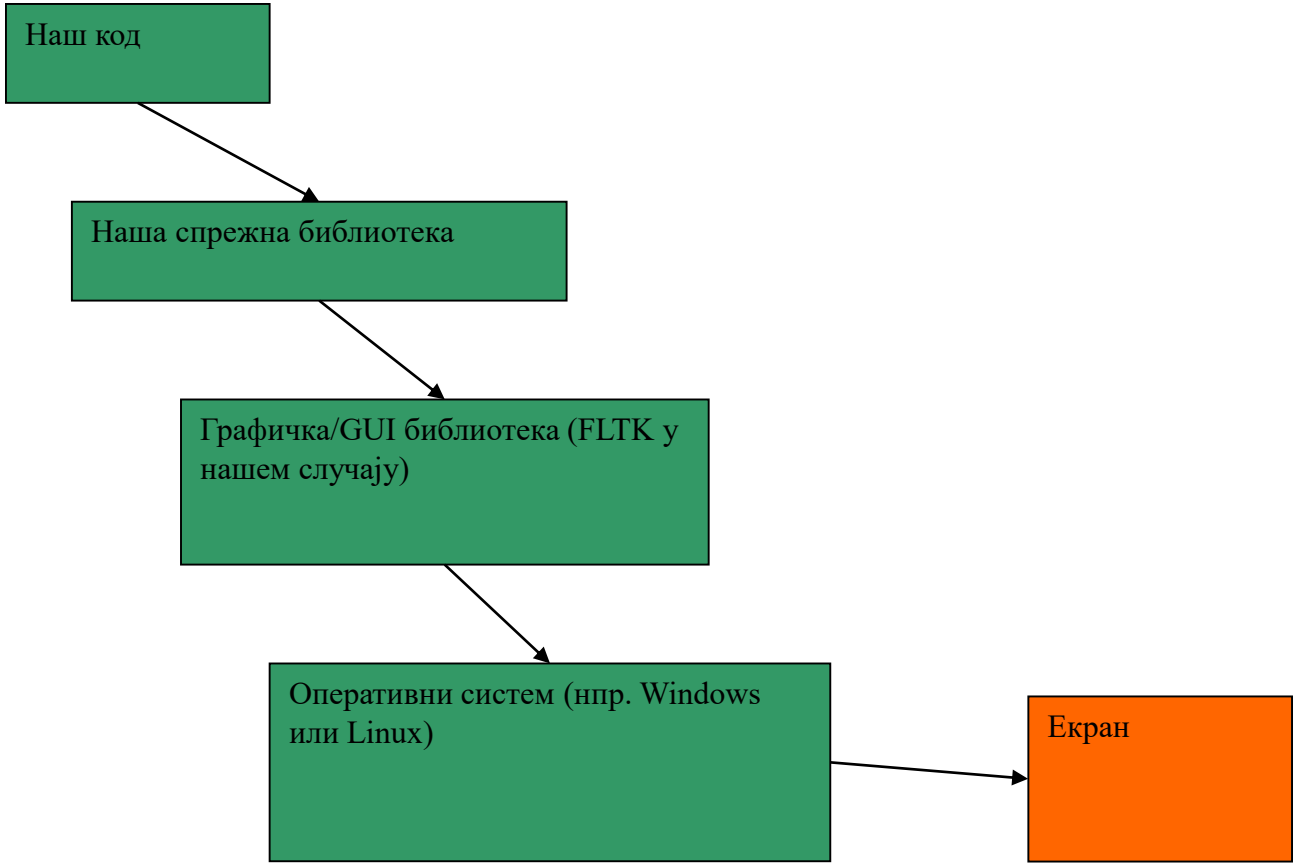
Графичка корисничка спрега
GUI

GUI

- Са програмерског становишта GUI се заснива на два концепта:
  - Објектно оријентисано програмирање
    - За организовање делова програма у складу са графичким појмовима и са заједничким спрегама и акцијама
  - Догађаји
    - За повезивање догађаја са програмским акцијама

Структура графичке спреге



- Слојевита архитектура

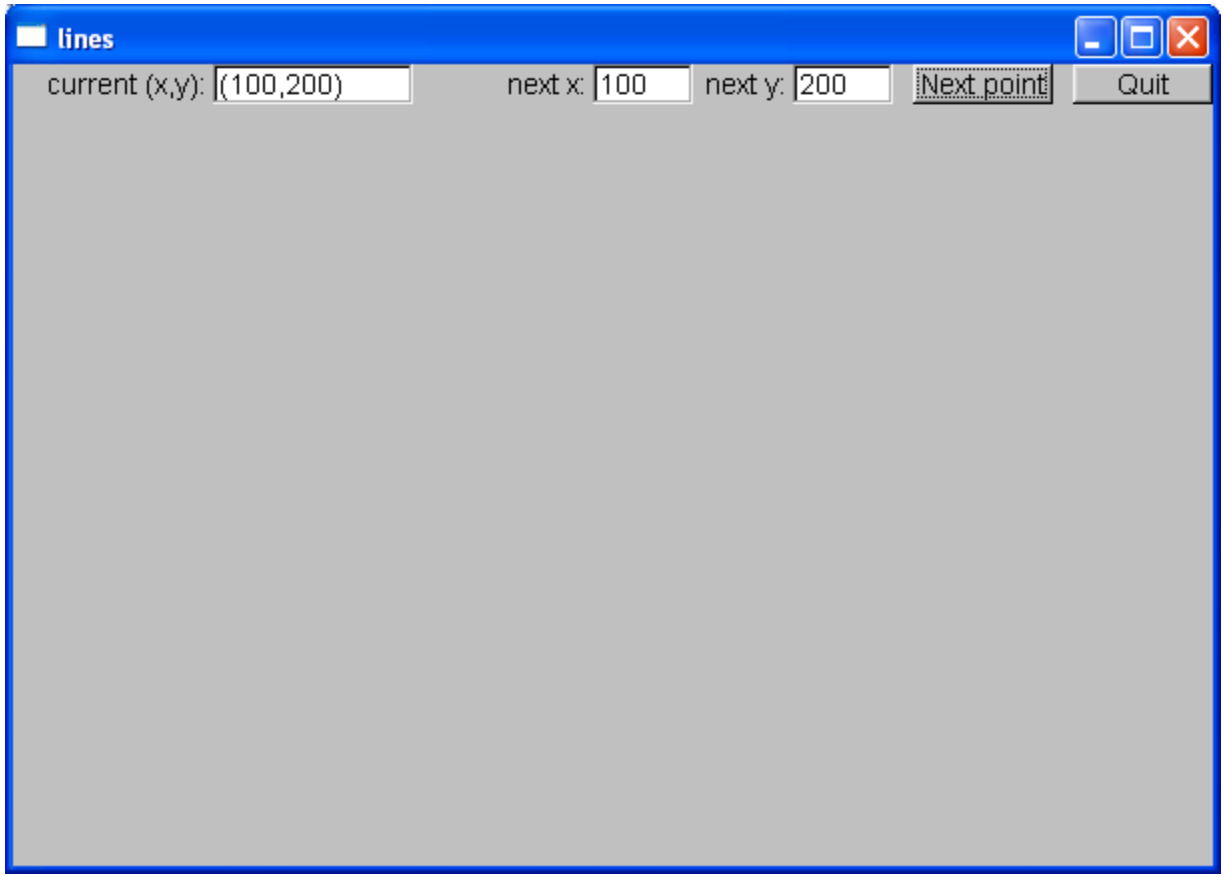
Пример

- Прозор са
  - два дугмета (**Button**), два улазна поља (**In_box**) и једно излазно поље (**Out_box**)

Пример

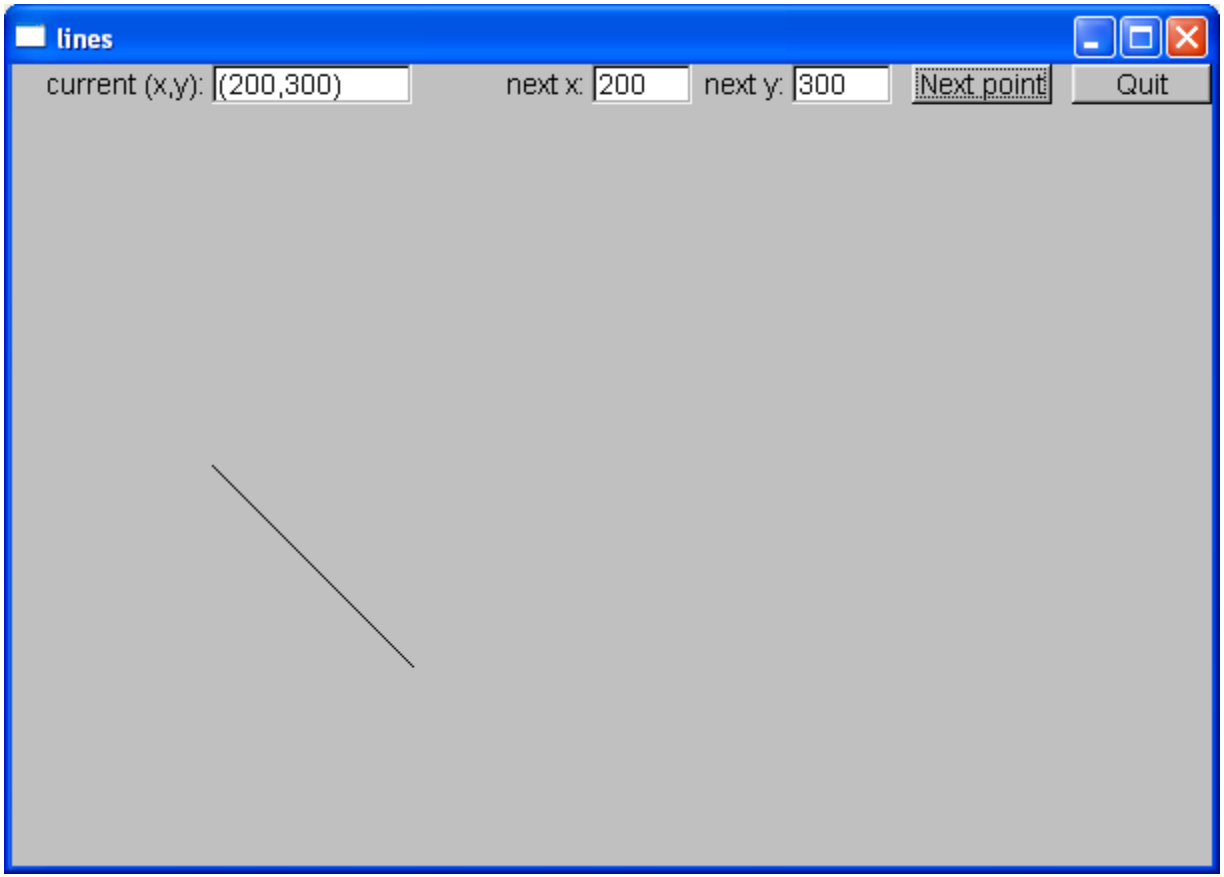
- Уносимо координате у улазна поља

Пример



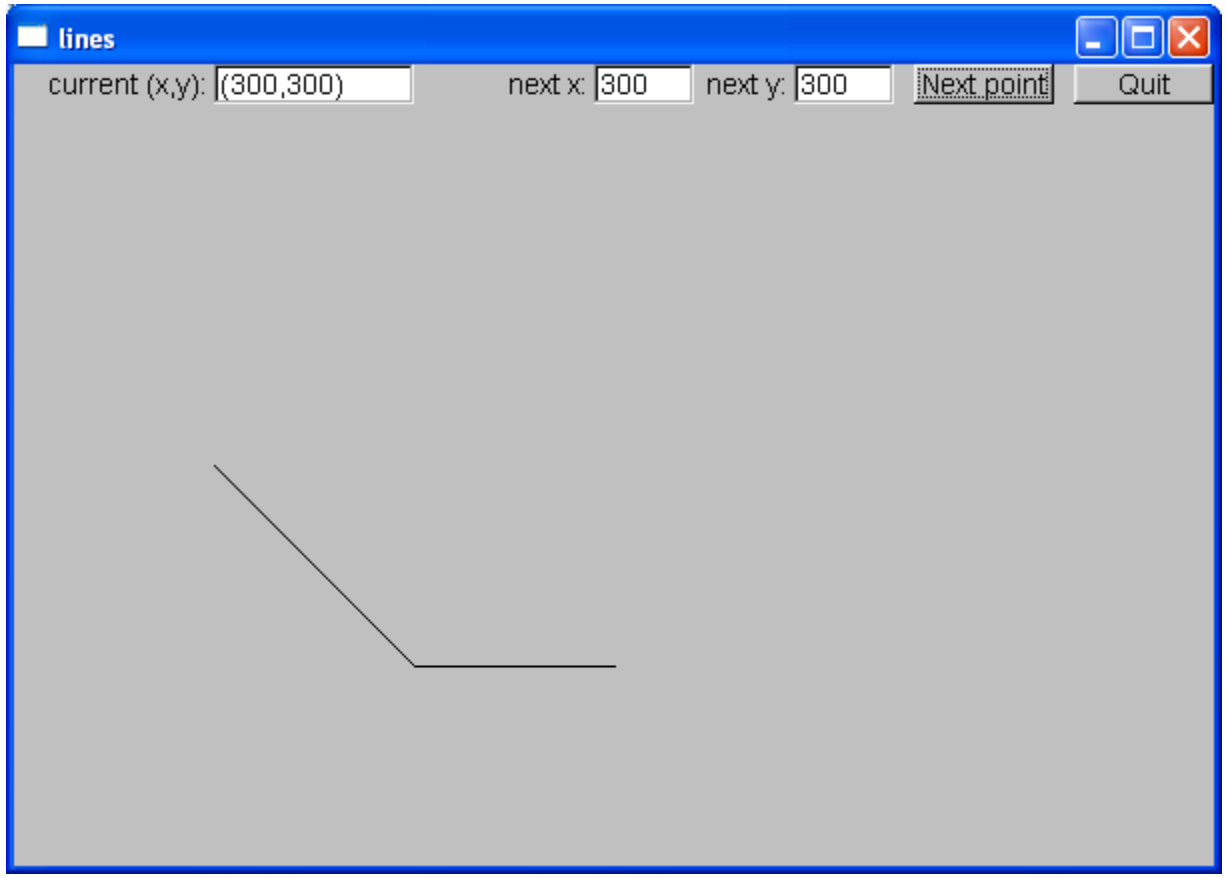
- Кад стиснемо дугме **Next point** унете координате се прикажу у излазном пољу.

Пример



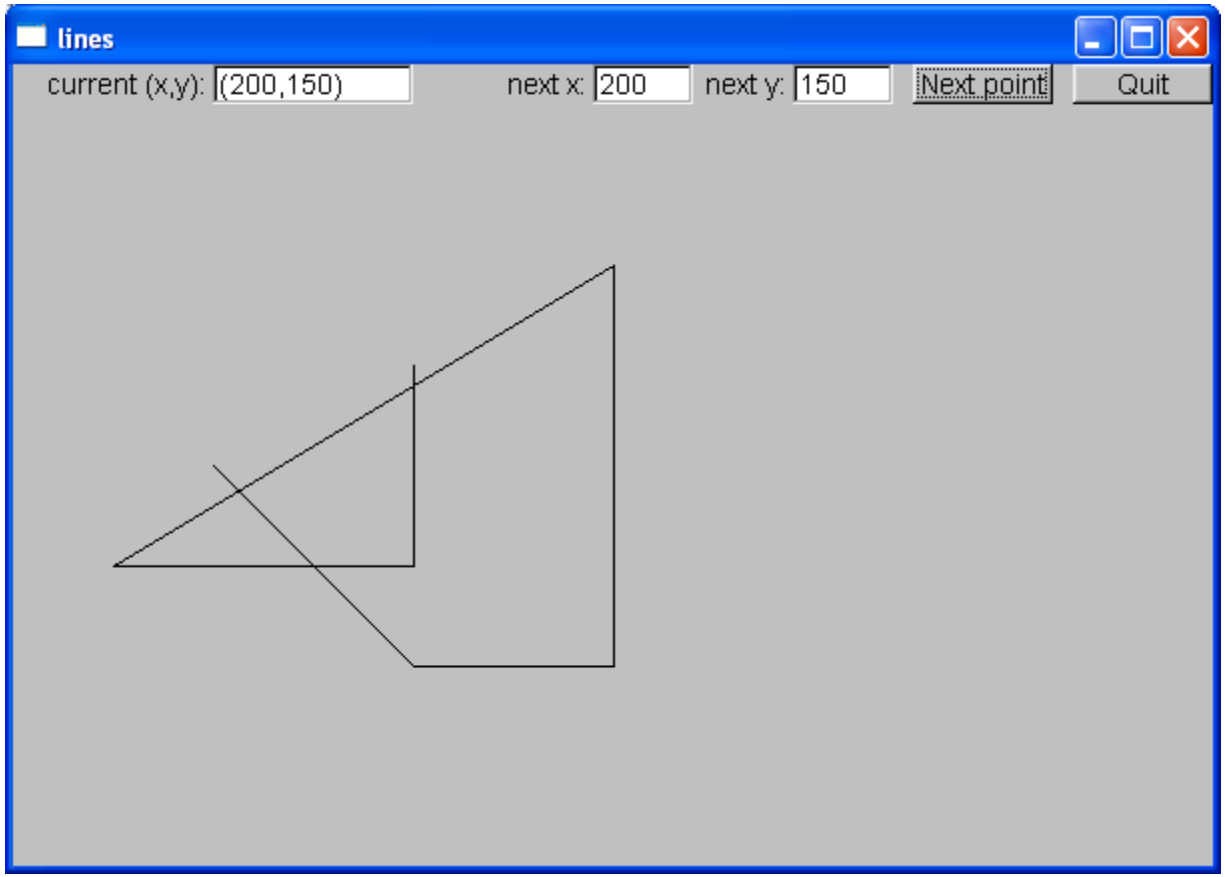
- Додајемо још једну тачку – образовали смо линију

Пример



- Након додавања треће тачке видимо да образујемо вишелинијски објекат (**Open_Polyline**)

Пример



- И тако све док не стиснемо дугме **Quit.**

Шта видимо и како се то дешава

- Имамо дугмад, улазна и излазна поља у прозору
  - Како формирамо прозор?
  - Како формирамо дугме?
  - Како формирамо улазна и излазна поља?
- Кликнемо на неко дугме и нешто се догоди
  - Како да испрограмирамо ту акцију?
  - Како да повежемо тај код са тим дугметом?
- Можемо да упишемо нешто у улазно поље
  - Како добављамо написани садржај?
- Видимо испис у излазном пољу
  - Како саопштавамо садржај који ту треба да се прикаже?
- Видимо линије које се исцртавају
  - Како памтимо линије?
  - Како их исцртавамо?

Наша спрежна библиотека

- Пресликавамо наше концепте на FTLK варијанту уобичајених графичких и GUI елемената.

Класа Lines_window

```
class Lines_window : public Window // Lines_window наслеђује Window
{
public:
  Lines_window(Point xy, int w, int h, const string& title);
  Open_polyline lines;

private:
  Button next_button;
  Button quit_button;
  In_box next_x;
  In_box next_y;
  Out_box xy_out;

  void next();  // акција када се next_button притисне
  void quit();  // акција када се quit_botton притисне

  static void cb_next(Address, Address window); // веза између дугмета и акције
  static void cb_quit(Address, Address window); // веза између дугмета и акције
};
```

Lines_window конструктор

```
Lines_window::Lines_window(Point xy, int w, int h, const string& title)
  : Window(xy,w,h,title),
    // конструктори за делове/елементе прозора:
    //        позиција            величина   назив       акција
  next_button(Point(x_max()-150,0), 70, 20, "Next point", cb_next),
  quit_button(Point(x_max()-70,0), 70, 20, "Quit", cb_quit),
  next_x(Point(x_max()-310,0), 50, 20, "next x:"),
  next_y(Point(x_max()-210,0), 50, 20, "next y:"),
  xy_out(Point(100,0), 100, 20, "current (x,y):")
{
  attach(next_button);
  attach(quit_button);
  attach(next_x);
  attach(next_y);
  attach(xy_out);
  attach(lines);
}
```

Виџити, Дугмад и Колбек функције

- Виџит (енгл. Widget) је објекат који је видљив и који има неку придружену акцију. У питању је генерализација свих активних графичких објеката, слично као што је Облик (енгл. Shape) генерализација свих пасивних графичких објеката.

- Рецимо, Дугме је Виџит који се приказује као правоугаоник са називом на средини, а када се клинке на њега треба да се деси нека акција.

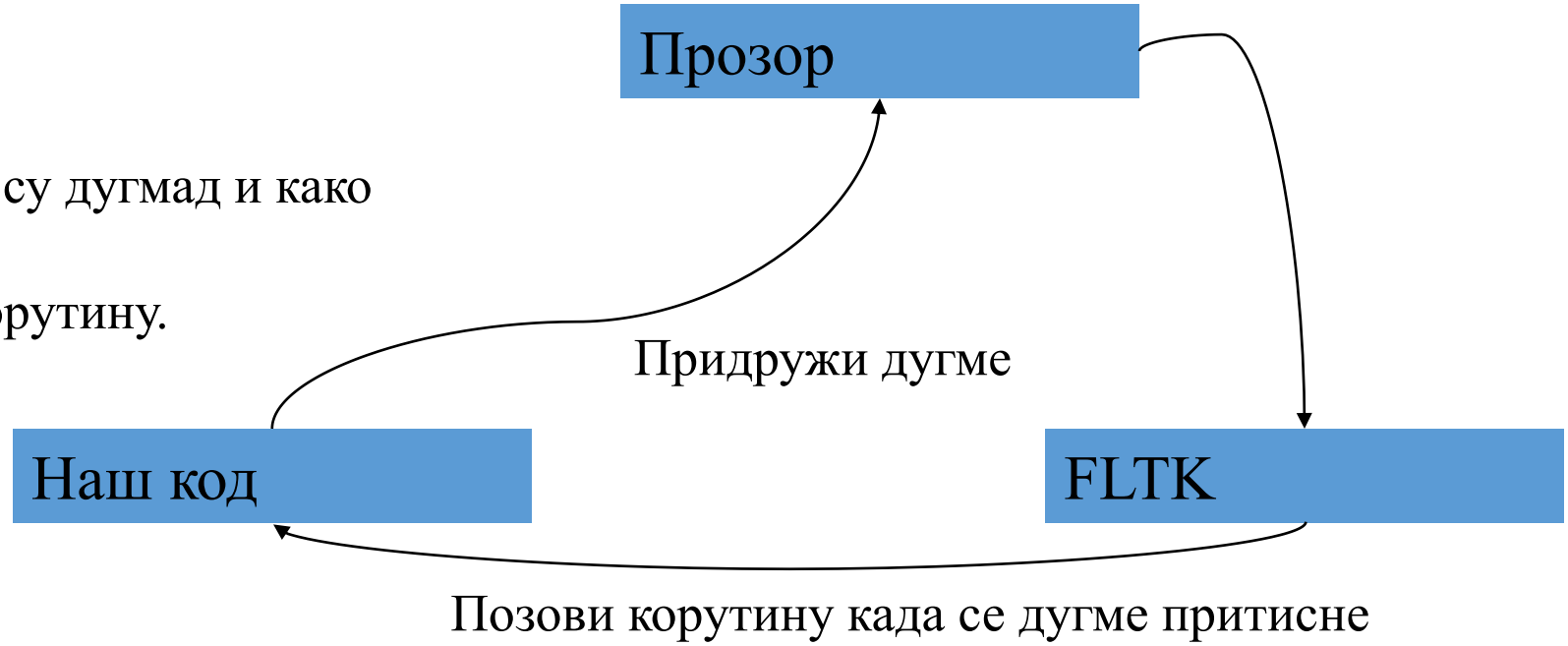
- Колбек (енгл. Callback) функција, тј. корутина, је функција која повезује догађај који се тиче дугмета са неком акцијом (обично описаном у посебној функцији)

Виџити, Дугмад и Колбек функције

```
class Button : Widget
{
public:
 Button(Point xy, int w, int h, const string& s, Callback cb)
     : Widget(xy, w, h, s, cb) { }
};
```

Како то ради?

Описује где су дугмад и како изгледају.
Региструј корутину.



Виџит

- Основни концепт на „Windows“ и „X windows“ системима
  - Практично све што видите на екрану је некакав виџит (Мајкрософт га зове „контрола“)

```
class Widget
{
public:
 Widget(Point xy, int w, int h, const string& s, Callback cb)
   : loc(xy), width(w), height(h), label(s), do_it(cb)
 { }
 // … веза са FLTK …
};
```

Дугме

- Дугме је Виџит који
  - се приказује како правоугаоник са називом
  - кад се кликне на њега позива се Колбек функција

```
class Button : Widget
{
public:
 Button(Point xy, int w, int h, const string& s, Callback cb)
     : Widget(xy, w, h, s, cb) { }
};
```

Колбек функција

- Колбек функције су део спреге ка „систему“
  - Повезивање функција и виџита је врло неелегантно у већини GUI-а
  - Не би морало бити тако, али
    - „систем“ у суштини не зна за Це++ (или било који виши програмски језик)
    - стил повезивања најчешће проистиче из подршке за Це, асемблер итд.
    - Већи системи се обично развијају у више програмских језика. Овакав механизам је један начин превазилажења разлика између језика.
  - Позив колбек функције се конвертује из системске позивне конвенције на ону која је одређена Це++ компајлером.

```
void Lines_window::cb_quit(Address, Address pw)
  // Позива Lines_window::quit() за објекат Прозор који се налази на адреси pw
{
  reference_to<Lines_window>(pw).quit();
}
```

Огољено:

```
void Lines_window::cb_quit(void*, void* pw)
  // Позива Lines_window::quit() за објекат Прозор који се налази на адреси pw
{
  static_cast<Lines_window*>(pw)->quit();
}
```

Акција quit

```
void Lines_window::quit()
{
  // можемо урадити било шта унутар објекта Lines_window
  hide();   // FLTK функција
}
```

Акција next

```
void Lines_window::next()
{
  int x = next_x.get_int();
  int y = next_y.get_int();

  lines.add(Point(x,y));

  stringstream ss;
  ss << '(' << x << ',' << y << ')';
  xy_out.put(ss.str());

  redraw();   // FLTK функција
}
```

In_box

```
class In_box : Widget
{
public:
  In_box(Point xy, int w, int h, const string& s)
       : Widget(xy, w, h, s, 0) { }
  int get_int();
  string get_string();
};

int In_box::get_int()
{
  // добављање референце на FLTK FL_Input вицит
  Fl_Input& pi = reference_to<Fl_Input>(pw);

  return atoi(pi.value());  // добави садржај
                            // и конвертуј из знакова у број (alpha to int)
}
```

Инверзија тока управљања

- Где је програм?
  - Наш програм само одговара на догађаје везане за разне виџите
  - Нема ниједне петље, ниједног гранања.
- Програм је напросто:

```
int main ()
{
  Lines_window win(Point(100,100), 600, 400, ”lines”);
  return gui_main();
}
```

Инверзија тока управљања