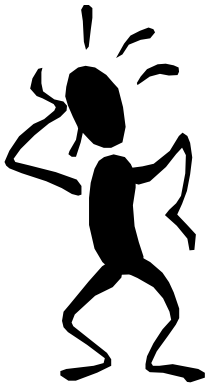
C++: что нужно знать, чтобы пройти интервью в Intel

Давным-давно

Основные принципы ООП

- Полиморфизм (абстракция)
- Инкапсуляция
- Наследование



Как было на C

```
enum Type
{
   Square,
   Circle,
   Triangle
};

int InitSquare(Object *pObj, ...);
int InitCircle(Object *pObj, ...);
int InitTriangle(Object *pObj, ...);

void DrawSquare(Object *pObj, ...);
void DrawCircle(Object *pObj, ...);
void DrawTriangle(Object *pObj, ...);

void CloseSquare(Object *pObj, ...);
void CloseCircle(Object *pObj, ...);
void CloseTriangle(Object *pObj, ...);
```

```
struct Object
{
   // Object type
   Type id;
   // Object parameters
   int a, b, c;
   // Handle
   Handle h;
};
```

Как было на C

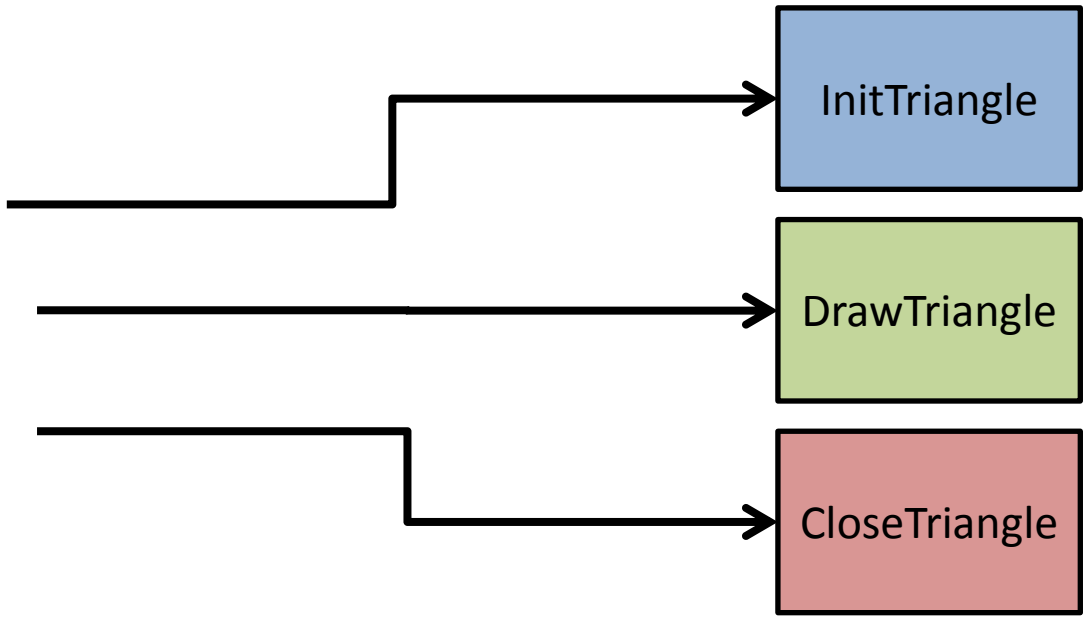
```
Object obj;
…
switch (obj.id)
{
   case Square:
      res = InitSquare(&obj, ...);
      break;
   case Circle:
      res = InitCircle(&obj, ...);
      break;
   case Triangle:
      res = InitTriangle(&obj, ...);
      break;
   default:
      break;
}
```

Как было на C

```
struct Object
{
   // Object type
   Type id;

   // Pointer to Init function
   int (*Init)(Object *pObj, …);
   // Pointer to Draw function
   void (*Draw)(Object *pObj, …);
   // Pointer to Close function
   void (*Close)(Object *pObj, …);

   // Object parameters
   int a, b, c;
   // Handle
   Handle h;
};
```

Как было на C

```
struct Object
{
   // Pointer to Init function
   int (*Init)(Object *pObj, …);
   // Pointer to Draw function
   void (*Draw)(Object *pObj, …);
   // Pointer to Close function
   void (*Close)(Object *pObj, …);

   // Object parameters
   int a, b, c;
   // Handle
   Handle h;
};
```



Как было на C

```
Object *pObj;
…

pObj = malloc(sizeof(Object));
PrepareFunctionPointers(pObj, Triangle);

res = pObj->Init(pObj, …);

pObj->Draw(pObj, …);

pObj->Close(pObj, …);
free(pObj);
```

Как было на C

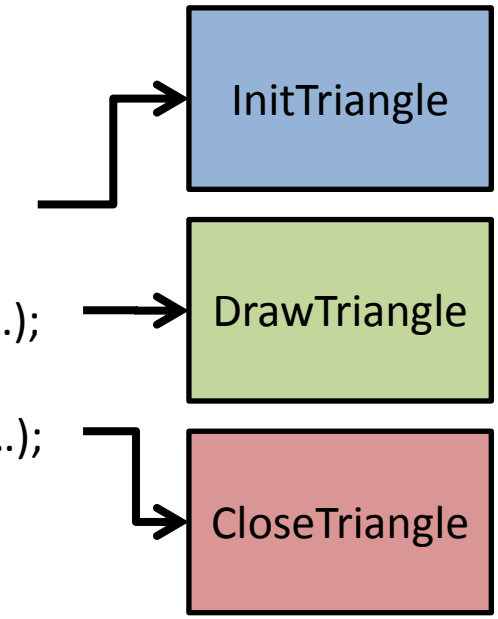
```
struct Object
{
   // Pointer to Init function
   int (*Init)(Object *pObj, …);
   // Pointer to Draw function
   void (*Draw)(Object *pObj, …);
   // Pointer to Close function
   void (*Close)(Object *pObj, …);

   // Pointer to function table
   const FuncTable *table;

   // Object parameters
   int a, b, c;
   // Handle
   Handle h;
};
```

```
struct FuncTable
{
   // Pointer to Init function
   int (*Init)(Object *pObj, …);
   // Pointer to Draw function
   void (*Draw)(Object *pObj, …);
   // Pointer to Close function
   void (*Close)(Object *pObj, …);
};
```

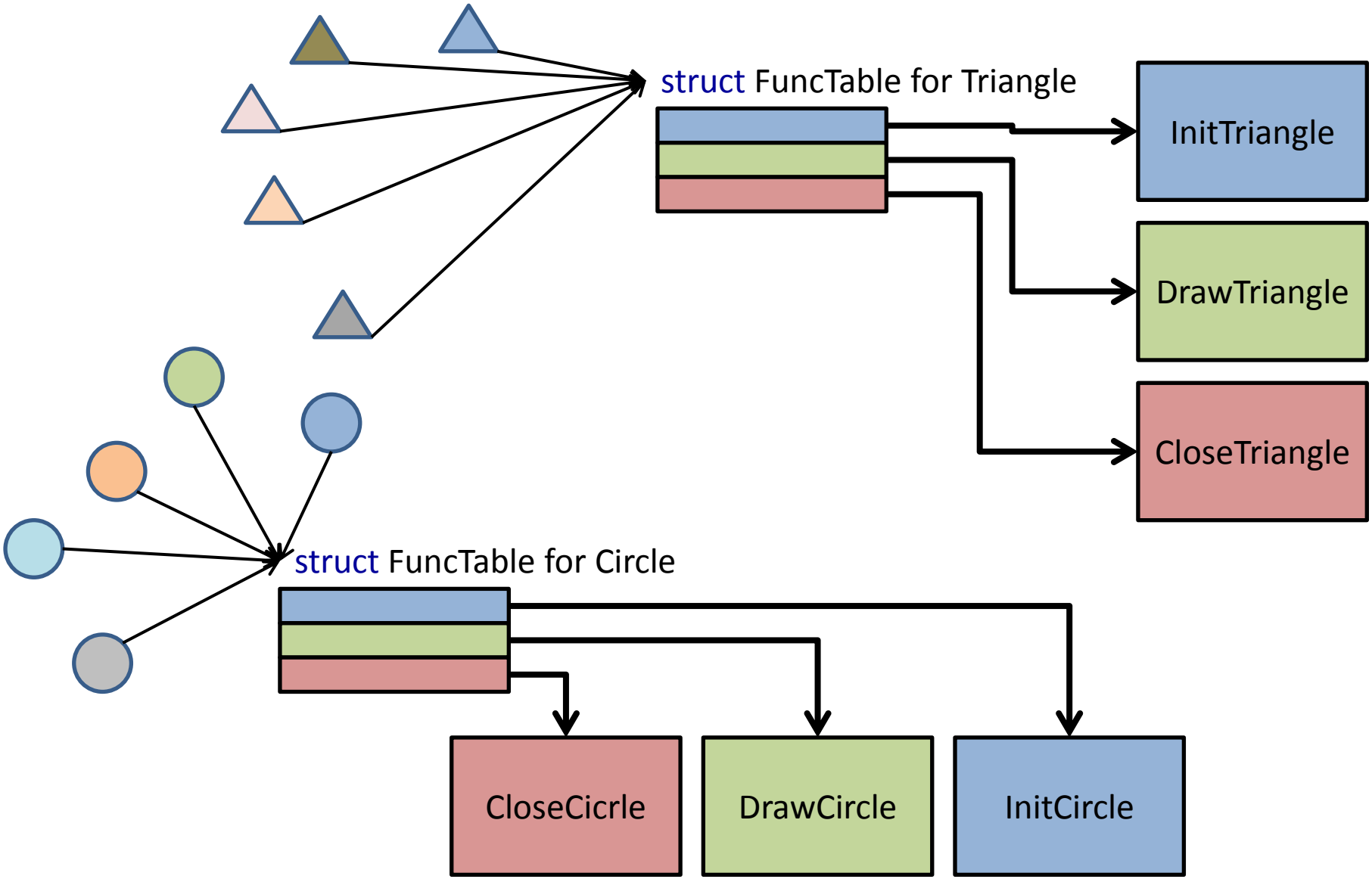
Как было на C

```
struct Object
{
   // Pointer to function table
   const FuncTable *table;

   // Object parameters
   int a, b, c;
   // Handle
   Handle h;
};
```

```
struct FuncTable
{
   // Pointer to Init function
   int (*Init)(Object *pObj, …);
   // Pointer to Draw function
   void (*Draw)(Object *pObj, …);
   // Pointer to Close function
   void (*Close)(Object *pObj, …);
};
```



Как было на C


struct FuncTable for Triangle
InitTriangle
DrawTriangle
CloseTriangle
struct FuncTable for Circle
CloseCicrle
DrawCircle
InitCircle

Как было на C

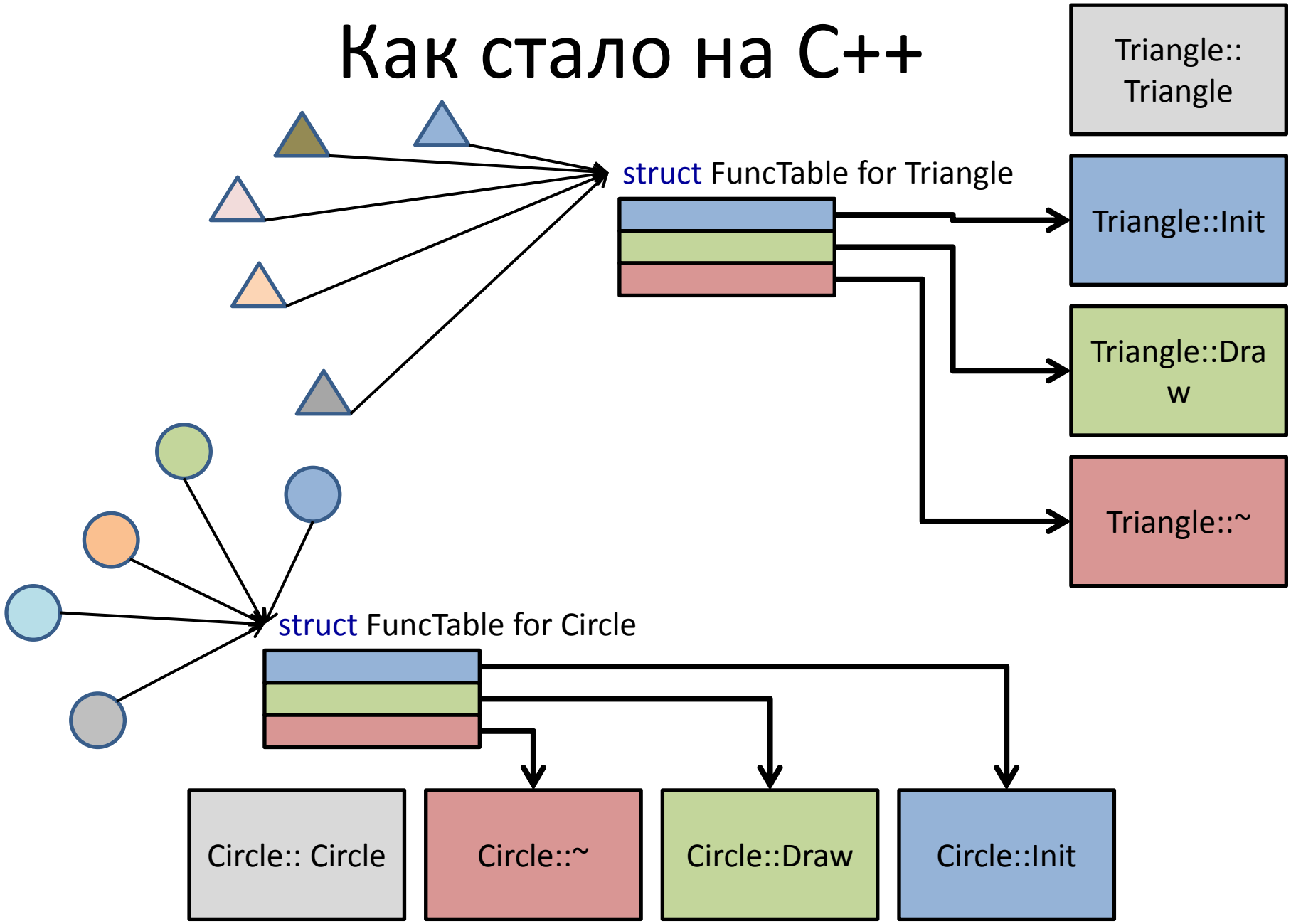
```
Object *pObj;
…

pObj = malloc(sizeof(Object));
PrepareFunctionPointers(pObj, Triangle);

res = pObj->Init(pObj, …);

pObj->Draw(pObj, …);

pObj->Close(pObj, …);
free(pObj);
```

Как было и как стало

```
Object *pObj;
…

pObj = malloc(sizeof(Object));
PrepareFunctionPointers(pObj, Triangle);

res = pObj->table->Init(pObj, …);

pObj->table->Draw(pObj, …);

pObj->table->Close(pObj, …);
free(pObj);
```

```
Object *pObj;
…

pObj = new Triangle;

res = pObj->Init(…);

pObj->Draw(…);

delete pObj;
```

Как стало на C++



Как было и как стало

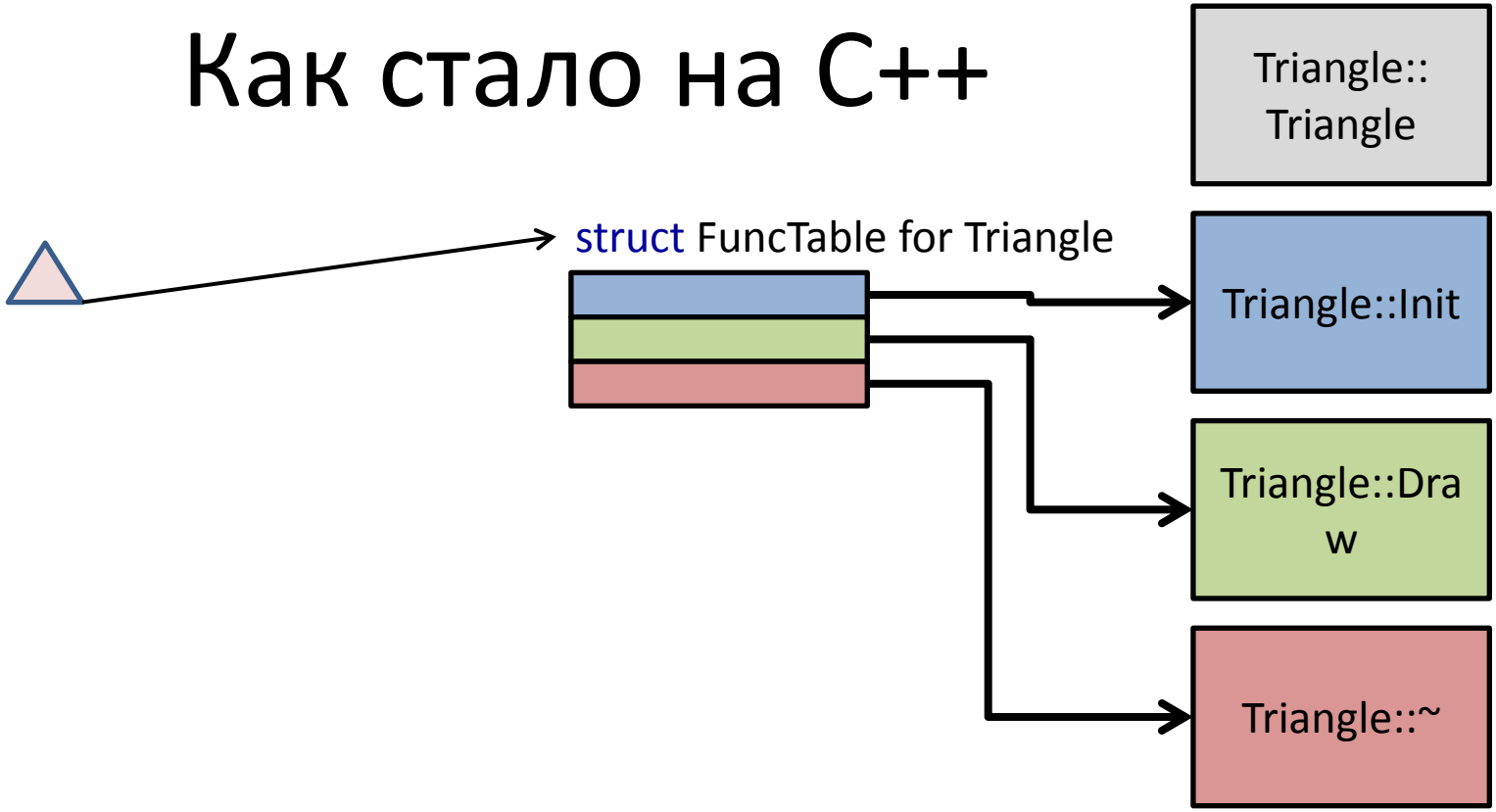
```
Object *pObj;
…

pObj = malloc(sizeof(Object));
PrepareFunctionPointers(pObj, Triangle);

res = pObj->table->Init(pObj, …);

pObj->table->Draw(pObj, …);

pObj->table->Close(pObj, …);
free(pObj);
```

```
Object *pObj;
…

pObj = new Triangle;

res = pObj->Init(…);

pObj->Draw(…);

delete pObj;
```

Как стало на C++

```
pObj = new Triangle;
pObj = malloc(sizeof(Triangle);
pObj->table = &FuncTableTriangle;
mov ecx, [pObj]
call Triangle::Triangle();
```

Как стало на C++



Как было и как стало

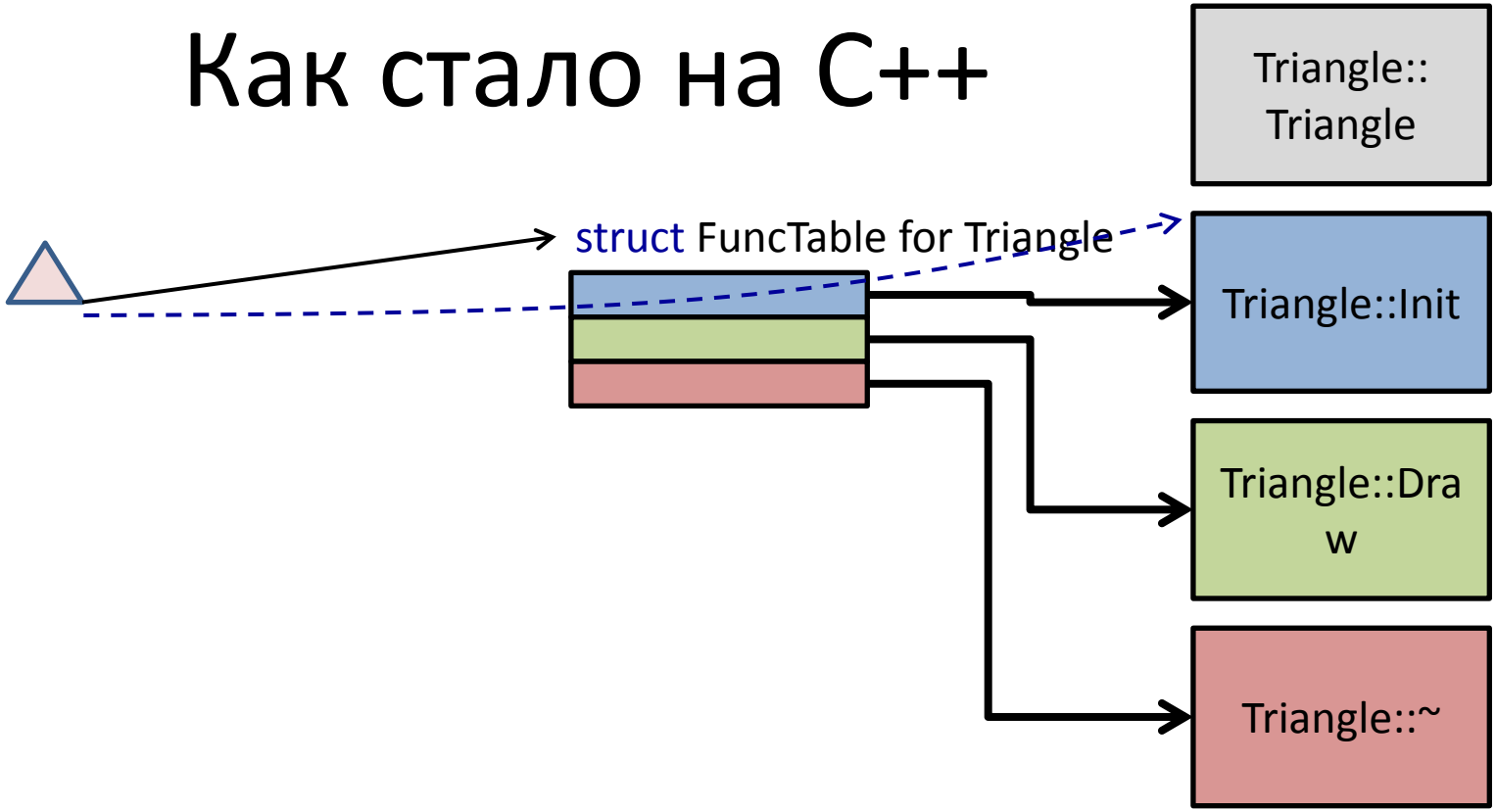
```
Object *pObj;
…

pObj = malloc(sizeof(Object));
PrepareFunctionPointers(pObj, Triangle);

res = pObj->table->Init(pObj, …);

pObj->table->Draw(pObj, …);

pObj->table->Close(pObj, …);
free(pObj);
```

```
Object *pObj;
…

pObj = new Triangle;

res = pObj->Init(…);

pObj->Draw(…);

delete pObj;
```

Как стало на C++

res = pObj->Init(…);
mov ecx, [pObj]
call pObj->table->Init(…)

Как стало на C++



Как было и как стало

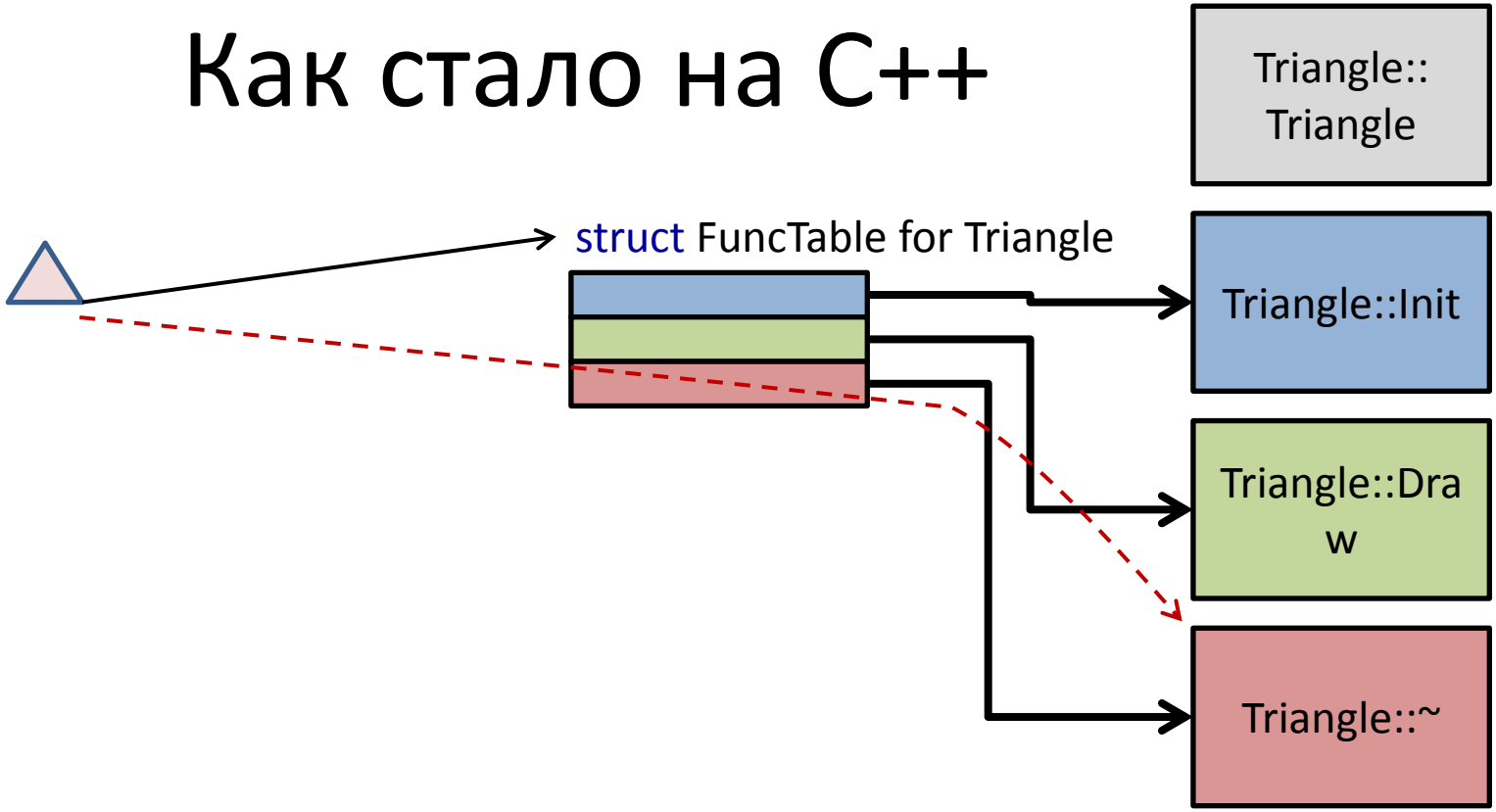
```
Object *pObj;
…

pObj = malloc(sizeof(Object));
PrepareFunctionPointers(pObj, Triangle);

res = pObj->table->Init(pObj, …);

pObj->table->Draw(pObj, …);

pObj->table->Close(pObj, …);
free(pObj);
```

```
Object *pObj;
…

pObj = new Triangle;

res = pObj->Init(…);

pObj->Draw(…);

delete pObj;
```

Как стало на C++

delete pObj;
mov ecx, [pObj]
call pObj->table->~Triangle()
free(pObj)

Как стало на C++



Как стало на C++

```
Object *pObj;
…

pObj = new Triangle;


res = pObj->Init(…);

pObj->Draw(…);

delete pObj;
```

Как стало на C++

```
class Square
{
public:
   // Constructor
   Square(void);
   // Destructor

   ~Square (void);
   // Initialize the object

   res Init(…);
   // Draw the object

   void Draw(…);
   // Close the object

   void Close(…);
};
```

```
class Circle
{
public:
   // Constructor
   Circle(void);
   // Destructor

   ~Circle(void);
   // Initialize the object

   res Init(…);
   // Draw the object

   void Draw(…);
   // Close the object

   void Close(…);
};
```

```
class Triangle
{
public:
   // Constructor
   Triangle(void);
   // Destructor

   ~Triangle(void);
   // Initialize the object

   res Init(…);
   // Draw the object

   void Draw(…);
   // Close the object

   void Close(…);
};
```

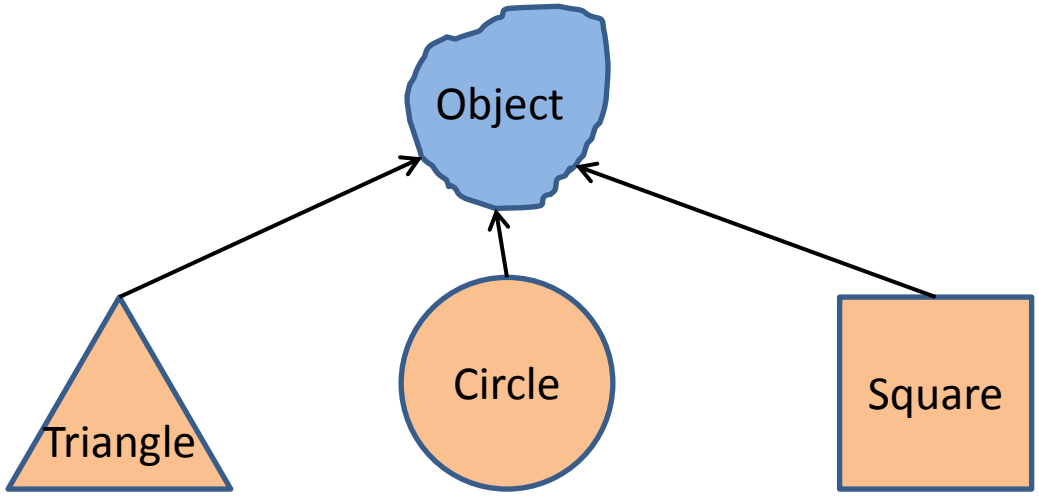
C++: наследование

```
class Object
{
public:

   // Destructor
   virtual
   ~Object(void) = 0;
   // Initialize the object
   virtual
   res Init(…) = 0;
   // Draw the object
   virtual
   void Draw(…) = 0;
   // Close the object
   virtual
   void Close(…) = 0;
};
```

```
class Circle
{
public:
   // Constructor
   Circle(void);
   // Destructor

   ~Circle(void);
   // Initialize the object

   res Init(…);
   // Draw the object

   void Draw(…);
   // Close the object

   void Close(…);
};
```

```
class Circle : public Object
{
public:
   // Constructor
   Circle(void);
   // Destructor
   virtual
   ~Circle(void);
   // Initialize the object
   virtual
   res Init(…);
   // Draw the object
   virtual
   void Draw(…);
   // Close the object
   virtual
   void Close(…);
};
```

C++: полиморфизм

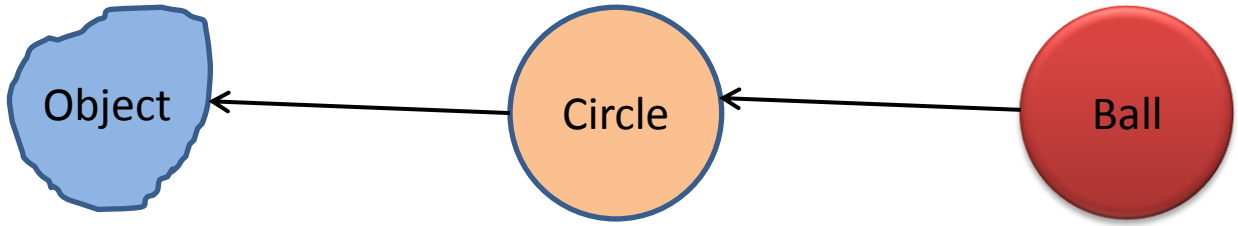



```
void PrintArea(Object *pObj)
{
    printf("Object area is %u\n", pObj->Area());
}
```

C++: инкапсуляция

```
Object *pObj;
…

pObj = CreateTriangle();

res = pObj->Init(…);

pObj->Draw(…);

delete pObj;
```

C++:наследование

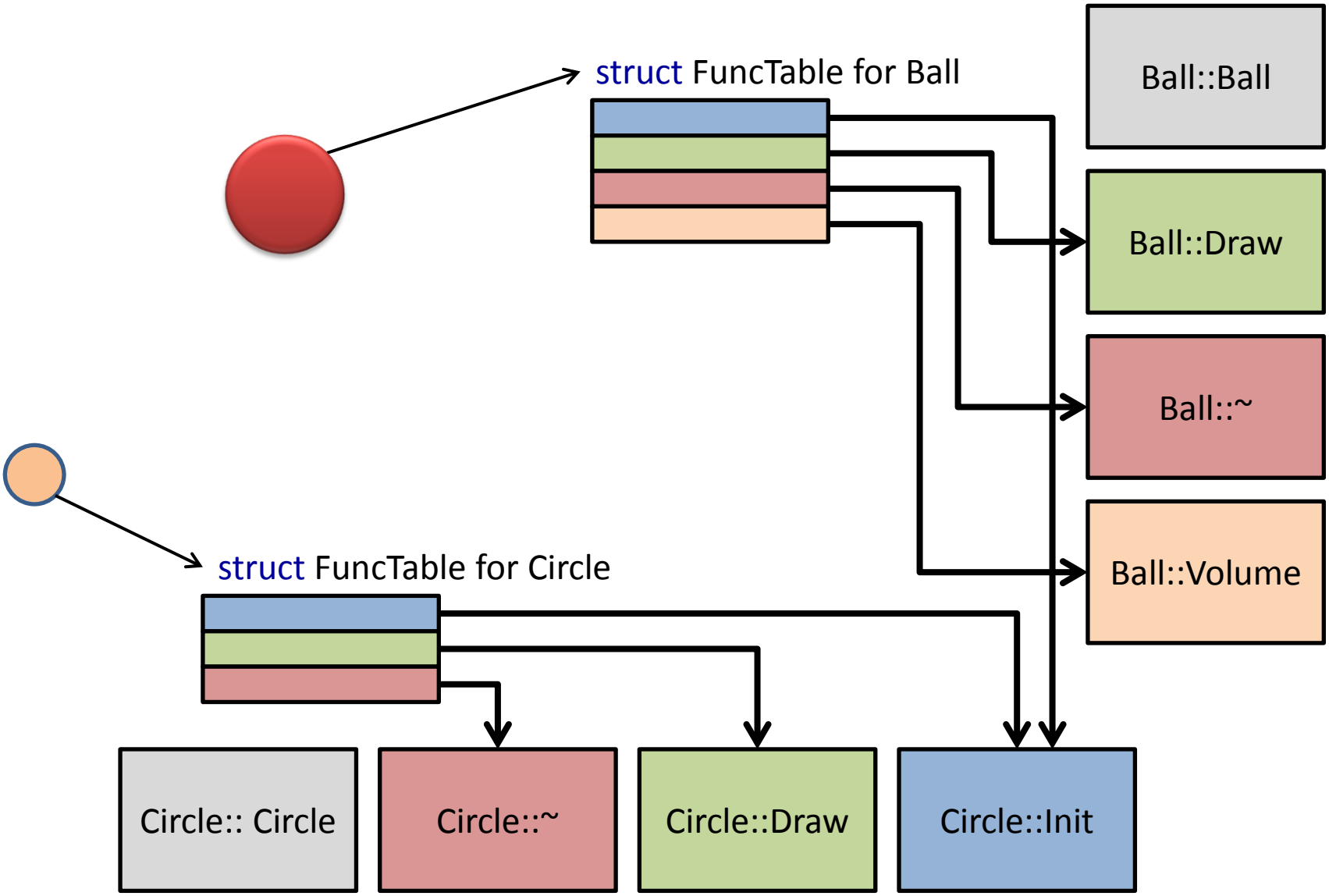



```
class Object
{
public:
   …
};
```

```
class Circle : public Object
{
public:
   …
};
```

```
class Ball : public Circle
{
public:
   // Destructor
   virtual
   ~Ball(void) ;
   // Draw the ball
   virtual
   void Draw(…);
   // Get the volume
   virtual
   int Volume(…);
};
```

C++:наследование



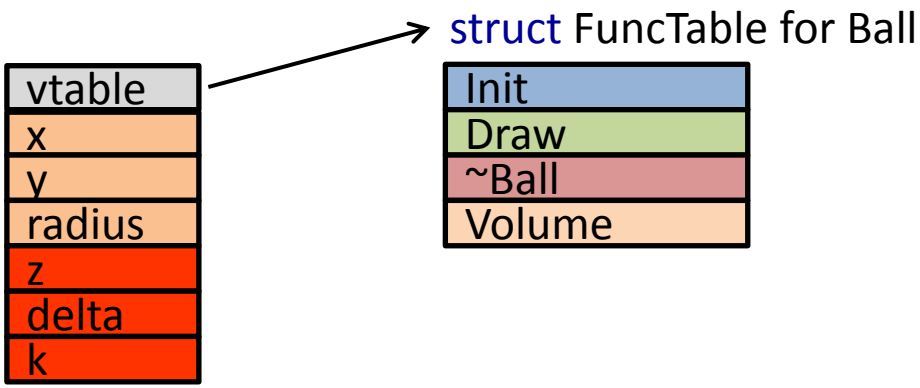
C++:наследование

```
class Object
{
public:

   // Destructor
   virtual
   ~Object(void) = 0;
   // Initialize the object
   virtual
   res Init(…) = 0;
   // Draw the object
   virtual
   void Draw(…) = 0;
   // Close the object
   virtual
   void Close(…) = 0;
};
```

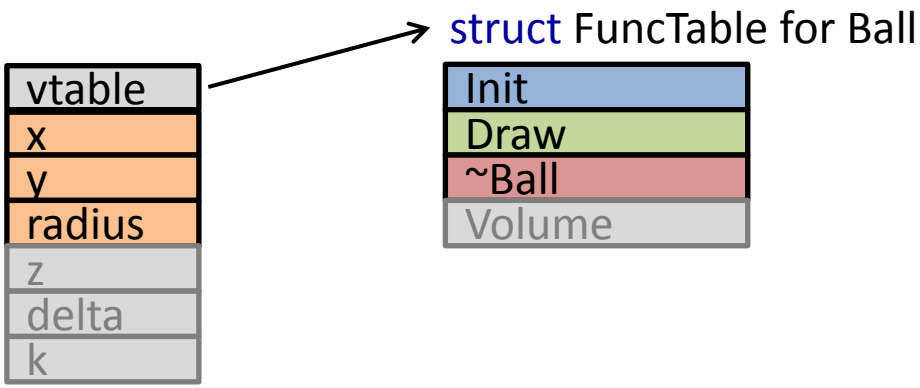
```
class Circle : public Object
{
public:
   // Constructor
   Circle(void);

protected:
   int x, y;
   int radius;
};
```

```
class Ball : public Circle
{
public:
   // Constructor
   Ball(void);

protected:
   int z;
   float delta;
   int k;
};
```

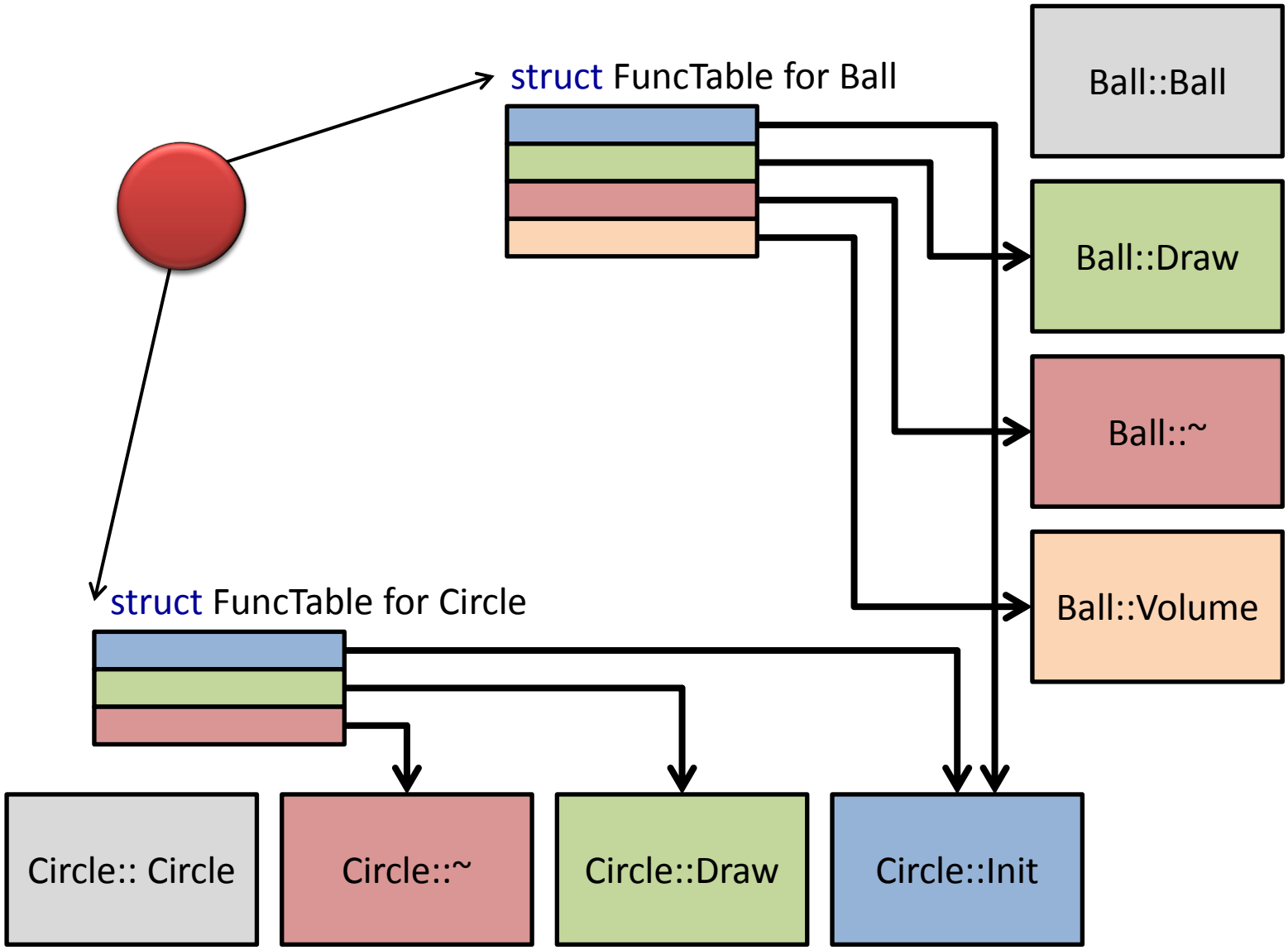
C++:наследование



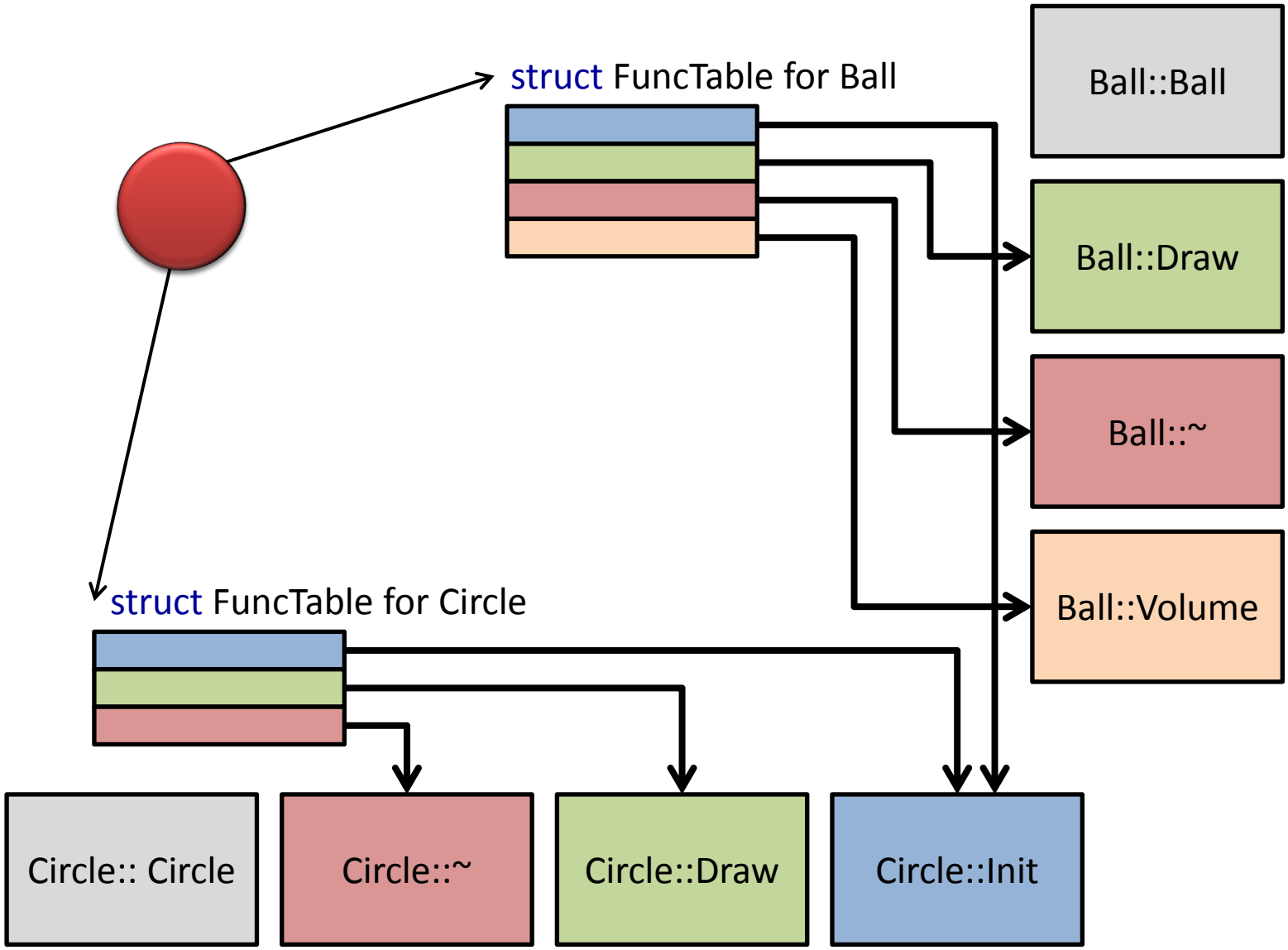
C++:наследование



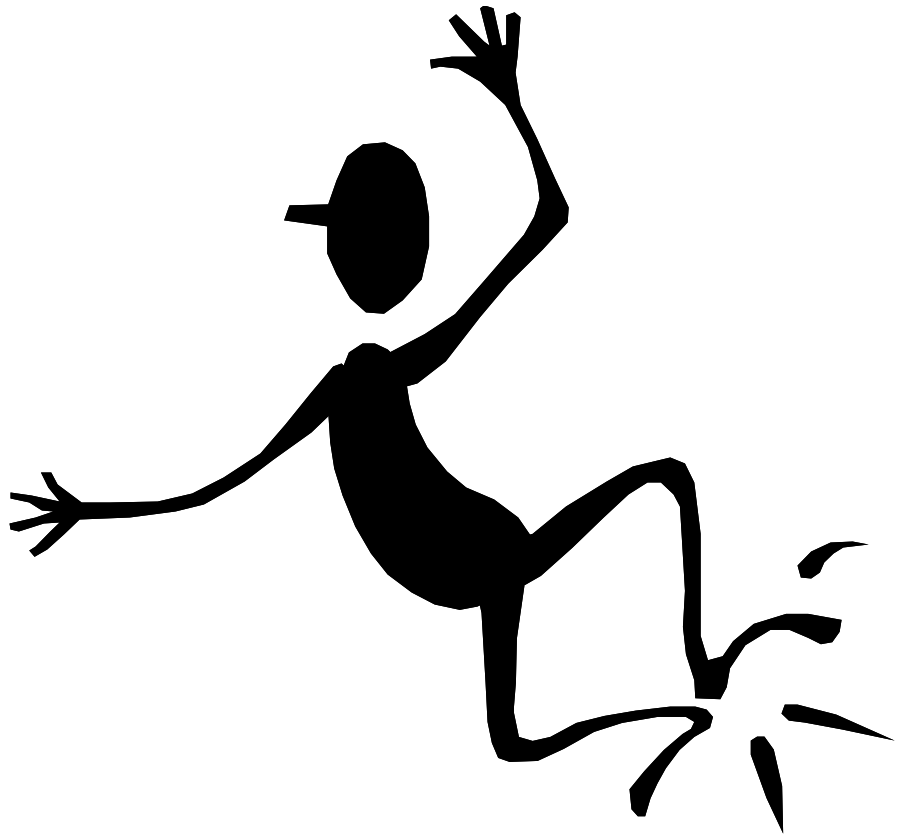
C++: конструкция объекта



C++: деструкция объекта



C++: ничего сложного



Пожелания

- Учите ассемблер
- Пишите часто на C и C++ для себя
- Думайте, как бы реализовали это вы