

Using Class Libraries

- **C++ is extensible**
- **Add new data types**
 - With new operations
- **Class library creation**
 - Individual programmer (CS-183)
 - Other developer(s)

What is a Class?

- **Programmer-defined type**
 - Similar to built-in type
 - Attributes
 - Operations
- **Operation forms**
 - Operators (like built-in)
 - “Messages” (member functions)

Class Libraries & Reuse

- **General purpose classes**
 - Domain-specific
 - Cross-domain
- **“Build” versus “buy”**
 - Build your own from scratch?
 - Use solid libraries instead?

Member Functions

- Also known as “messages”
- Like “normal” functions
 - Name
 - Return value type
 - Arguments
- Applied to a particular object

Member Function Example

```

cout << setw(8) << 3 << endl;
      ↑
Manipulator
      ↓
      Member function
      ↓
      cout.width(8);
      ↓
cout << 3 << endl;

```

Member Function Prototypes

- Similar to “normal” functions
 - Name, return type, arguments
- Often in header file
 - Referenced by #include
- Enclosed in class declaration

Member Function Prototypes

```

class Loan ← Class definition
{
public:
    void SetPrincipal(double pr);
    void SetInterest(double rate);
    void SetTerm(int periods);
    double GetPayment(); ← Prototypes
}

```

Member Function Usage

```

Loan mtg; ← Object definition
// Mortgage loan
// Set loan characteristics
mtg.SetPrincipal(100000.0);
mtg.SetInterest(0.085);
mtg.SetTerm(30*12);
// Output loan payment
cout << mtg.GetPayment(); ← Function calls

```

How Do Class Objects Work?

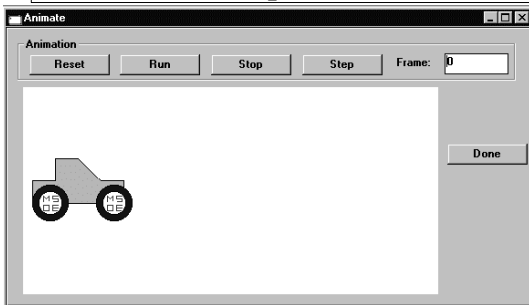
- **Attributes**
 - Internal data (usually private)
 - Accessed by operations
- **Behaviors**
 - Member functions
 - Operators

More detail
in CS-183!

Animation Example

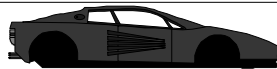
- **Drawing area class**
 - Provides "window" or "canvas"
 - Manages a set of graphic objects
- **Drawing object classes**
 - Common behaviors
 - Unique appearance

Animation Example



Drawing Object Classes

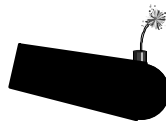
Car body



Car wheel



Cannon



Cannon ball



Also other shapes; see lab handout.

⋮

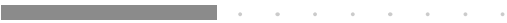
Drawing Object Functions (1)

```

int GetXPosition() const;
int GetYPosition() const;
void SetPosition(int x, int y);
void Move(int dx, int dy);

double GetRotation() const;
void SetRotation(double ang);
void Rotate(double delta_angle);

```



Handwritten notes area with horizontal lines.

⋮

Drawing Object Functions (2)

```

unsigned int GetSize() const;
void SetSize(unsigned int sz);

void SetColor(int r,
              int g,
              int b);
// Red, green, blue primaries
// 0 .. 255 for each value

```



Handwritten notes area with horizontal lines.

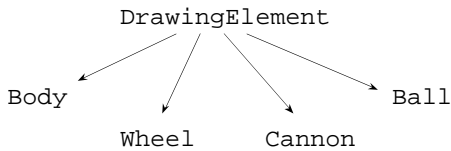
⋮

Drawing Area Function

```

void AddElement
(DrawingElement& elem);

```



Handwritten notes area with horizontal lines.

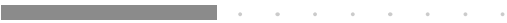
⋮

Animation Framework Calls (1)

```
unsigned int SetupAnimation
    (CDrawArea& draw_area);
```

One of the functions you must write.

Called once at beginning of program.
Used to define drawing elements (global objects) in animation.
AddElement member function of CDrawArea.



⋮

Animation Framework Calls (2)

```
void ResetAnimation();
```

Called each time animation is restarted.
Used to initialize drawing elements (positions, sizes, colors, etc.).
Calls member functions of drawing element objects.



⋮

Animation Framework Calls (3)

```
void DoNextFrame
    (unsigned int frame);
```

Called once for each "frame" in animation.
Used to modify drawing elements (positions, sizes, colors, etc.).
Calls member functions of drawing element objects.