The Art of Identifier Naming

Michael John Decker
Adapted with permission from Jonathan I. Maletic
Kent State University

Names

- Variables
- Types, classes
- Methods, functions
- Global constants

The most difficult part of programming is coming up with good variable names.

Describe this code:

```
int f(int x, int y) {

  int z;

  z = x * y;

  return z;

}
```

Describe this code:

```
int f(int h, int w) {

  int a;

  a = h * w;

  return a;

}
```

Describe this code:

```
int area_of_rectangle(int height, int width) {

  int area;

  area = height * width;

  return area;

}
```

Nouns

- Variables, objects, types, classes
  - person, stack, list, window, menu
- Plurals should be avoided
  - string people[n]; // people[3]="joe";
  - string person[n]; // person[3]="joe";

Verbs

- functions, methods
- get_value, push, compute_salary
- draw_window, update
- size_of, add, is_equal

Naming style

- Consistent style should be used
- camelCase - capitalize words (after the first)
- under_score - underscores between words.

Constants

- Constants are always all caps and underscore separated
- Examples:
  - MAX_DIGITS
  - VERSION_NUMBER

Indentation & Spacing

- Use consistent indentation and spacing
- Most IDEs use 4 spaces per block
- Take advantage and use the auto indent
- Take time to go back and clean up indentation
- Avoid mixing tabs and spaces

Ugly & Unreadable

```
  void foo(int x) {
int   t;
  for (int i=0;    i<  x; ++i) {
std::cout << i;
t =t+  i; }
  std::cout    <<  t;
   }
```

Clean & Neat

```
void foo(int x) {
    int t;
    for (int i = 0; i < x; ++i) {
        std::cout << i;
        t = t +  i;
    }
     std::cout <<  t;
}
```