

```
1. #
2. # Makefile
3. #
4. # David J. Malan
5. # malan@harvard.edu
6. #
7.
8. all: bounce button checkbox click cursor label slider text window
9.
10. bounce: bounce.c Makefile
11.     clang -ggdb3 -O0 -std=c99 -Wall -Werror -o bounce bounce.c -lcs -lm
12.
13. button: button.c Makefile
14.     clang -ggdb3 -O0 -std=c99 -Wall -Werror -o button button.c -lcs -lm
15.
16. checkbox: checkbox.c Makefile
17.     clang -ggdb3 -O0 -std=c99 -Wall -Werror -o checkbox checkbox.c -lcs -lm
18.
19. click: click.c Makefile
20.     clang -ggdb3 -O0 -std=c99 -Wall -Werror -Wno-unused-variable -o click click.c -lcs -lm
21.
22. cursor: cursor.c Makefile
23.     clang -ggdb3 -O0 -std=c99 -Wall -Werror -o cursor cursor.c -lcs -lm
24.
25. label: label.c Makefile
26.     clang -ggdb3 -O0 -std=c99 -Wall -Werror -o label label.c -lcs -lm
27.
28. slider: slider.c Makefile
29.     clang -ggdb3 -O0 -std=c99 -Wall -Werror -o slider slider.c -lcs -lm
30.
31. text: text.c Makefile
32.     clang -ggdb3 -O0 -std=c99 -Wall -Werror -o text text.c -lcs -lm
33.
34. window: window.c Makefile
35.     clang -ggdb3 -O0 -std=c99 -Wall -Werror -o window window.c -lcs -lm
36.
37. clean:
38.     rm -f *.o core bounce button checkbox click cursor label slider text window
```

```
1. /**
2.  * bounce.c
3.  *
4.  * David J. Malan
5.  * malan@harvard.edu
6.  *
7.  * Bounces a circle back and forth in a window.
8.  */
9.
10. // standard libraries
11. #include <stdio.h>
12.
13. // Stanford Portable Library
14. #include <spl/gevents.h>
15. #include <spl/gobjects.h>
16. #include <spl/gwindow.h>
17.
18. int main(void)
19. {
20.     // instantiate window
21.     GWindow window = newGWindow(320, 240);
22.
23.     // instantiate circle
24.     GOval circle = newGOval(0, 110, 20, 20);
25.     setColor(circle, "BLACK");
26.     setFilled(circle, true);
27.     add(window, circle);
28.
29.     // initial velocity
30.     double velocity = 2.0;
31.
32.     // bounce forever
33.     while (true)
34.     {
35.         // move circle along x-axis
36.         move(circle, velocity, 0);
37.
38.         // bounce off right edge of window
39.         if (getX(circle) + getWidth(circle) >= getWidth(window))
40.         {
41.             velocity = -velocity;
42.         }
43.
44.         // bounce off left edge of window
45.         else if (getX(circle) <= 0)
46.         {
47.             velocity = -velocity;
48.         }

```

---

```
49.  
50.     // linger before moving again  
51.     pause(10);  
52. }  
53. }
```

```
1. /**
2.  * button.c
3.  *
4.  * David J. Malan
5.  * malan@harvard.edu
6.  *
7.  * Demonstrates a button.
8.  */
9.
10. // standard libraries
11. #include <stdio.h>
12. #include <string.h>
13.
14. // Stanford Portable Library
15. #include <spl/gevents.h>
16. #include <spl/ginteractors.h>
17. #include <spl/gwindow.h>
18.
19. int main(void)
20. {
21.     // instantiate window
22.     GWindow window = newGWindow(320, 240);
23.
24.     // instantiate button
25.     GButton button = newGButton("Button");
26.     setActionCommand(button, "click");
27.
28.     // add button to southern region of window
29.     addToRegion(window, button, "SOUTH");
30.
31.     // listen for events
32.     while (true)
33.     {
34.         // wait for event
35.         GActionEvent event = waitForEvent(ACTION_EVENT);
36.
37.         // if window was closed
38.         if (getEventType(event) == WINDOW_CLOSED)
39.         {
40.             break;
41.         }
42.
43.         // if action command is "click"
44.         if (strcmp(getActionCommand(event), "click") == 0)
45.         {
46.             printf("button was clicked\n");
47.         }
48.     }
```

---

```
49.  
50.     // that's all folks  
51.     closeGWindow(window);  
52.     return 0;  
53. }
```

```
1. /**
2.  * checkbox.c
3.  *
4.  * David J. Malan
5.  * malan@harvard.edu
6.  *
7.  * Demonstrates a checkbox.
8.  */
9.
10. // standard libraries
11. #include <stdio.h>
12. #include <string.h>
13.
14. // Stanford Portable Library
15. #include <spl/gevents.h>
16. #include <spl/ginteractors.h>
17. #include <spl/gwindow.h>
18.
19. int main(void)
20. {
21.     // instantiate window
22.     GWindow window = newGWindow(320, 240);
23.
24.     // instantiate checkbox
25.     GCheckBox checkbox = newGCheckBox("I agree");
26.     setActionCommand(checkbox, "check");
27.     addToRegion(window, checkbox, "SOUTH");
28.
29.     // listen for events
30.     while (true)
31.     {
32.         // wait for event
33.         GActionEvent event = waitForEvent(ACTION_EVENT);
34.
35.         // if window was closed
36.         if (getEventType(event) == WINDOW_CLOSED)
37.         {
38.             break;
39.         }
40.
41.         // if action command is "check"
42.         if (strcmp(getActionCommand(event), "check") == 0)
43.         {
44.             if (isSelected(checkbox))
45.             {
46.                 printf("checkbox was checked\n");
47.             }
48.             else
```

```
49.     {
50.         printf("checkbox was unchecked\n");
51.     }
52. }
53. }
54.
55. // that's all folks
56. closeGWindow(window);
57. return 0;
58. }
```

```
1. /**
2.  * click.c
3.  *
4.  * David J. Malan
5.  * malan@harvard.edu
6.  *
7.  * Prints cursor's location when mouse's button is clicked.
8.  */
9.
10. // standard libraries
11. #include <stdio.h>
12.
13. // Stanford Portable Library
14. #include <spl/gevents.h>
15. #include <spl/gwindow.h>
16.
17. int main(void)
18. {
19.     // instantiate window
20.     GWindow window = newGWindow(320, 240);
21.
22.     // listen forever
23.     while (true)
24.     {
25.         // check for mouse event
26.         GEvent event = getNextEvent(MOUSE_EVENT);
27.
28.         // if we heard one
29.         if (event != NULL)
30.         {
31.             // if the event was movement
32.             if (getEventType(event) == MOUSE_CLICKED)
33.             {
34.                 // print click's coordinates
35.                 printf("%.0f,%.0f\n", getX(event), getY(event));
36.             }
37.         }
38.     }
39. }
```



```
1. /**
2.  * cursor.c
3.  *
4.  * David J. Malan
5.  * malan@harvard.edu
6.  *
7.  * Draws a circle that follows user's cursor within a window.
8.  */
9.
10. // Stanford Portable Library
11. #include <spl/gevents.h>
12. #include <spl/gobjects.h>
13. #include <spl/gwindow.h>
14.
15. int main(void)
16. {
17.     // instantiate window
18.     GWindow window = newGWindow(320, 240);
19.
20.     // instantiate circle
21.     GOval circle = newGOval(0, 0, 50, 50);
22.
23.     // add circle to window
24.     add(window, circle);
25.
26.     // follow mouse forever
27.     while (true)
28.     {
29.         // check for mouse event
30.         GEvent event = getNextEvent(MOUSE_EVENT);
31.
32.         // if we heard one
33.         if (event != NULL)
34.         {
35.             // if the event was movement
36.             if (getEventType(event) == MOUSE_MOVED)
37.             {
38.                 // ensure circle follows top cursor
39.                 double x = getX(event) - getWidth(circle) / 2;
40.                 double y = getY(event) - getWidth(circle);
41.                 setLocation(circle, x, y);
42.             }
43.         }
44.     }
45. }
```

```
1. /**
2.  * label.c
3.  *
4.  * David J. Malan
5.  * malan@harvard.edu
6.  *
7.  * Draws a label in a window.
8.  */
9.
10. // standard libraries
11. #include <ctype.h>
12.
13. // Stanford Portable Library
14. #include <spl/gobjects.h>
15. #include <spl/gwindow.h>
16.
17. int main(void)
18. {
19.     // instantiate window
20.     GWindow window = newGWindow(320, 240);
21.
22.     // instantiate label
23.     GLabel label = newGLabel("");
24.     setFont(label, "SansSerif-36");
25.     add(window, label);
26.
27.     // count down from 50 to 0
28.     for (int i = 50; i >= 0; i--)
29.     {
30.         // to store 50 through 0 (with '\0'), we need <= 3 chars
31.         char s[3];
32.
33.         // convert i from int to string
34.         sprintf(s, "%i", i);
35.
36.         // update label
37.         setLabel(label, s);
38.
39.         // center label
40.         double x = (getWidth(window) - getWidth(label)) / 2;
41.         double y = (getHeight(window) - getHeight(label)) / 2;
42.         setLocation(label, x, y);
43.
44.         // linger for 100 milliseconds
45.         pause(100);
46.     }
47.
48.     // that's all folks
```

---

```
49.     closeGWindow(window);  
50.     return 0;  
51. }
```

```
1. /**
2.  * noswap.c
3.  *
4.  * David J. Malan
5.  * malan@harvard.edu
6.  *
7.  * Should swap two variables' values, but doesn't! How come?
8.  */
9.
10. #include <stdio.h>
11.
12. void swap(int a, int b);
13.
14. int main(void)
15. {
16.     int x = 1;
17.     int y = 2;
18.
19.     printf("x is %i\n", x);
20.     printf("y is %i\n", y);
21.     printf("Swapping...\n");
22.     swap(x, y);
23.     printf("Swapped!\n");
24.     printf("x is %i\n", x);
25.     printf("y is %i\n", y);
26. }
27.
28. /**
29.  * Fails to swap arguments' values.
30.  */
31. void swap(int a, int b)
32. {
33.     int tmp = a;
34.     a = b;
35.     b = tmp;
36. }
```

```
1. /**
2.  * pointers.c
3.  *
4.  * David J. Malan
5.  * malan@harvard.edu
6.  *
7.  * Prints a given string one character per line.
8.  *
9.  * Demonstrates pointer arithmetic.
10. */
11.
12. #include <cs50.h>
13. #include <stdio.h>
14. #include <string.h>
15.
16. int main(void)
17. {
18.     // get line of text
19.     char* s = GetString();
20.     if (s == NULL)
21.     {
22.         return 1;
23.     }
24.
25.     // print string, one character per line
26.     for (int i = 0, n = strlen(s); i < n; i++)
27.     {
28.         printf("%c\n", *(s+i));
29.     }
30. }
```

```
1. /**
2.  * sigma-0.c
3.  *
4.  * David J. Malan
5.  * malan@harvard.edu
6.  *
7.  * Adds the numbers 1 through n.
8.  *
9.  * Demonstrates iteration.
10. */
11.
12. #include <cs50.h>
13. #include <stdio.h>
14.
15. // prototype
16. int sigma(int);
17.
18. int main(void)
19. {
20.     // ask user for a positive int
21.     int n;
22.     printf("Positive integer please: ");
23.     n = GetInt();
24.
25.     // compute sum of 1 through n
26.     int answer = sigma(n);
27.
28.     // report answer
29.     printf("%i\n", answer);
30. }
31.
32. /**
33.  * Returns sum of 1 through m; returns 0 if m is not positive.
34.  */
35. int sigma(int m)
36. {
37.     // avoid risk of infinite loop
38.     // return sum of 1 through m
39.     int sum = 0;
40.     for (int i = 1; i <= m; i++)
41.     {
42.         sum += i;
43.     }
44.     return sum;
45. }
```

```
1. /**
2.  * sigma-1.c
3.  *
4.  * David J. Malan
5.  * malan@harvard.edu
6.  *
7.  * Adds the numbers 1 through n.
8.  *
9.  * Demonstrates recursion.
10. */
11.
12. #include <cs50.h>
13. #include <stdio.h>
14.
15. // prototype
16. int sigma(int);
17.
18. int main(void)
19. {
20.     // ask user for a positive int
21.     int n;
22.     do
23.     {
24.         printf("Positive integer please: ");
25.         n = GetInt();
26.     }
27.     while (n < 1);
28.
29.     // compute sum of 1 through n
30.     int answer = sigma(n);
31.
32.     // report answer
33.     printf("%i\n", answer);
34. }
35.
36. /**
37.  * Returns sum of 1 through m; returns 0 if m is not positive.
38.  */
39. int sigma(int m)
40. {
41.     // base case
42.     if (m <= 0)
43.     {
44.         return 0;
45.     }
46.
47.     // recursive case
48.     else
```

---

```
49.     {
50.         return (m + sigma(m - 1));
51.     }
52. }
```



```
1. /**
2.  * slider.c
3.  *
4.  * David J. Malan
5.  * malan@harvard.edu
6.  *
7.  * Demonstrates a slider.
8.  */
9.
10. // standard libraries
11. #include <stdio.h>
12. #include <string.h>
13.
14. // Stanford Portable Library
15. #include <spl/gevents.h>
16. #include <spl/ginteractors.h>
17. #include <spl/gwindow.h>
18.
19. int main(void)
20. {
21.     // instantiate window
22.     GWindow window = newGWindow(320, 240);
23.
24.     // instantiate slider
25.     addToRegion(window, newGLabel("0"), "SOUTH");
26.     GSlider slider = newGSlider(0, 100, 50);
27.     setActionCommand(slider, "slide");
28.     addToRegion(window, slider, "SOUTH");
29.     addToRegion(window, newGLabel("100"), "SOUTH");
30.
31.     // listen for events
32.     while (true)
33.     {
34.         // wait for event
35.         GActionEvent event = waitForEvent(ACTION_EVENT);
36.
37.         // if window was closed
38.         if (getEventType(event) == WINDOW_CLOSED)
39.         {
40.             break;
41.         }
42.
43.         // if action command is "slide"
44.         if (strcmp(getActionCommand(event), "slide") == 0)
45.         {
46.             printf("slider was slid to %i\n", getValue(slider));
47.         }
48.     }
```

---

```
49.  
50.     // that's all folks  
51.     closeGWindow(window);  
52.     return 0;  
53. }
```

```
1. /**
2.  * text.c
3.  *
4.  * David J. Malan
5.  * malan@harvard.edu
6.  *
7.  * Demonstrates a text field.
8.  */
9.
10. // standard libraries
11. #include <stdio.h>
12. #include <string.h>
13.
14. // Stanford Portable Library
15. #include <spl/gevents.h>
16. #include <spl/ginteractors.h>
17. #include <spl/gwindow.h>
18.
19. int main(void)
20. {
21.     // instantiate window
22.     GWindow window = newGWindow(320, 240);
23.
24.     // instantiate text field (wide enough for 10 characters)
25.     GTextField field = newGTextField(10);
26.     setActionCommand(field, "input");
27.     addToRegion(window, field, "SOUTH");
28.
29.     // listen for events
30.     while (true)
31.     {
32.         // wait for event
33.         GActionEvent event = waitForEvent(ACTION_EVENT);
34.
35.         // if window was closed
36.         if (getEventType(event) == WINDOW_CLOSED)
37.         {
38.             break;
39.         }
40.
41.         // if action command is "input"
42.         if (strcmp(getActionCommand(event), "input") == 0)
43.         {
44.             printf("\n%s\n was inputted\n", getText(field));
45.         }
46.     }
47.
48.     // that's all folks
```

---

```
49.     closeGWindow(window);  
50.     return 0;  
51. }
```

```
1. /**
2.  * window.c
3.  *
4.  * David J. Malan
5.  * malan@harvard.edu
6.  *
7.  * Draws a window.
8.  */
9.
10. // Stanford Portable Library
11. #include <spl/gwindow.h>
12.
13. int main(void)
14. {
15.     // instantiate window
16.     GWindow window = newGWindow(320, 240);
17.
18.     // let's look at it for a while
19.     pause(5000);
20.
21.     // that's all folks
22.     closeGWindow(window);
23.     return 0;
24. }
```