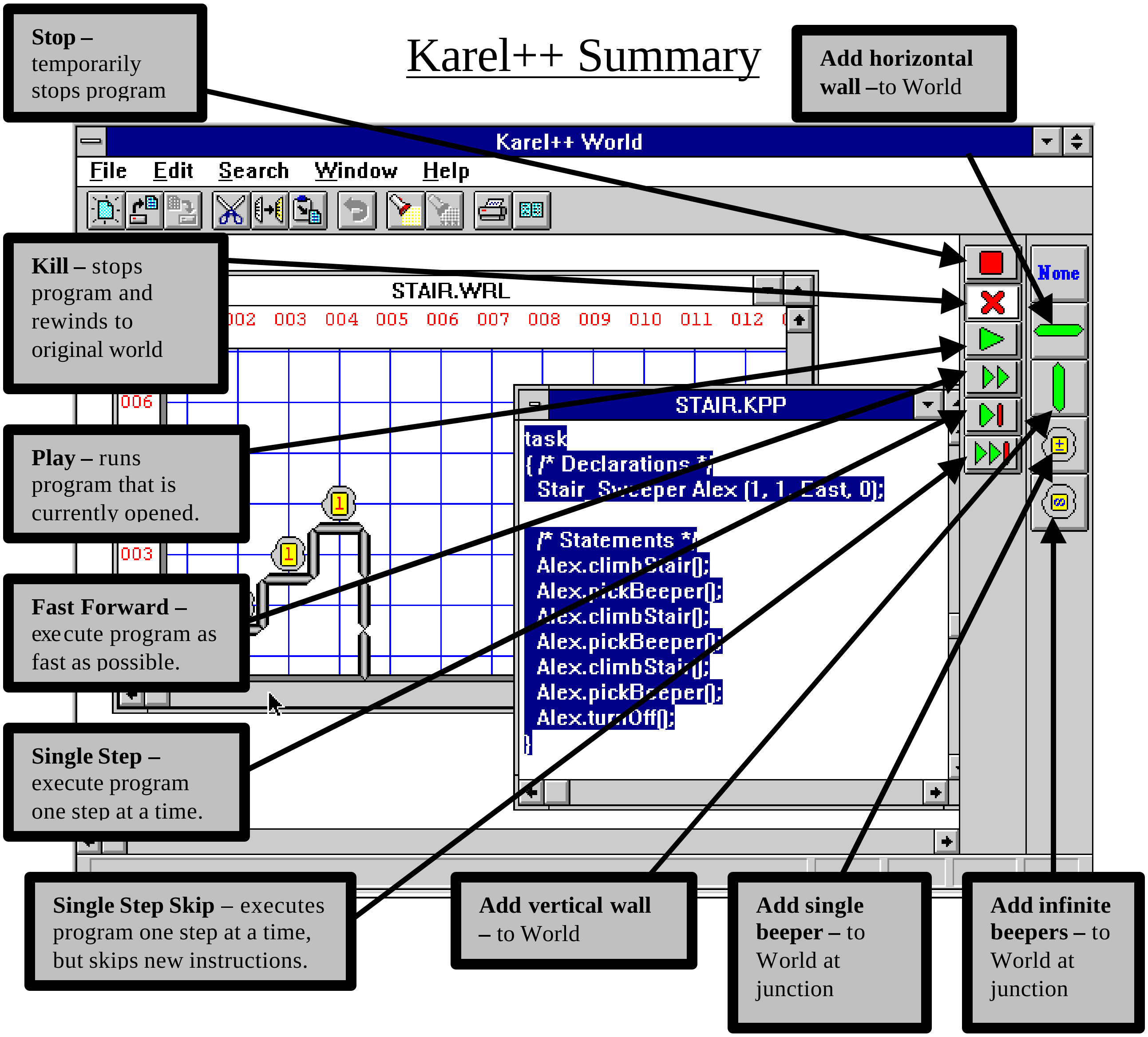
Karel++ Summary



```
task
{ /* Declarations */
  Stair_Sweeper Alex (1, 1, East, 0);

  /* Statements */
  Alex.climbStair();
  Alex.pickBeeper();
  Alex.climbStair();
  Alex.pickBeeper();
  Alex.climbStair();
  Alex.pickBeeper();
  Alex.turnOff();
}
```

Stop – temporarily stops program

Add horizontal wall –to World

Kill – stops program and rewinds to original world

Play – runs program that is currently opened.

Fast Forward – execute program as fast as possible.

Single Step – execute program one step at a time.

Single Step Skip – executes program one step at a time, but skips new instructions.

Add vertical wall – to World

Add single beeper – to World at junction

Add infinite beepers – to World at junction

Steps

1. Create a World by choosing *new* from the *file* menu and then using the pallet to add walls and beepers to the World.
2. Run Notepad and type in a program using the Karel++ language.
3. Save program as a .kpp file.
4. Open your Karel++ program using *open* from the Karel++ *file* menu and choosing the *file type* as "*Karel++ Source*", and then choosing your .kpp file.
5. Hit the play button (see above diagram).
6. Karel++ will report syntax errors. Fix the errors in Notepad then save the changes.
7. Repeat Steps 4, 5 and 6 until the program runs.

Primitive Operations

```
class ur_Robot
    {
        void move();       // robot moves forward one block

        void turnOff();    // robot turns itself off

        void turnLeft();   // robot pivots in place 90 degrees

        void pickBeeper(); // robot picks up beep and puts
it into a bag

        void putBeeper();  // robot takes beeper out of bag
and places on corner
    }
```

Conditions (tests)

```
    class Robot: ur_Robot
    {
      Boolean frontIsClear();

      Boolean nextToABeeper();

      Boolean nextToARobot();

      Boolean facingNorth();

      Boolean facingSouth();

      Boolean facingEast();

      Boolean facingWest();

      Boolean anyBeepersInBeeperBag();
    }
```

Robot Initialization

```
ur_Robot
<name>(<street>,<avenue>,<direction>,<numBeepers>);

Robot
<name>(<street>,<avenue>,<direction>,<numBeepers>);
```

Program Form

```
task
    {
       <robotInitialization>;
       ...
       <robotInitialization>;
       <instruction>;
       ...
       <instruction>;
    }
```

Conditional Instructions

```
if ( <test> )
        {
           <instruction>;
           ...
           <instruction>;
        }
-------------------------------------------
 if ( <test> )
        {
           <instruction>;
           ...
           <instruction>;
        }
 else
        {
           <instruction>;
           ...
           <instruction>;
        }
```

Repetitive (loop) Instructions

```
while ( <test> )
        {
           <instruction>;
           ...
           <instruction>;
        }
--------------------------------------------------
 loop ( <positiveNumber> )
        {
           <instruction>;
           ...
           <instruction>;
        }
```

Defining New Classes

```
class <new-class-name>: <old-class-name>
       {
       <list-of-instruction names>
       }

       <definition-of-new-instructions>
```

Defining New Instructions

```
void <instructions-class> :: <instruction-name>()
        {
           <instruction>;
           ...
           <instruction>;
        }
```

Errors

- A robot executes a move instruction when a wall blocks the path to the immediate next corner.
- A robot executes a pickBeeper when it is on a corner with no beepers.
- A robot executes a putBeeper when it has no beepers in its bag