Code bloat

In computer programming, **code bloat** is the production of program code (source code or machine code) that is perceived as unnecessarily long, slow, or otherwise wasteful of resources. Code bloat can be caused by inadequacies in the programming language in which the code is written, the compiler used to compile it, or the programmer writing it. Thus, while code bloat generally refers to source code size (as produced by the programmer), it can be used to refer instead to the *generated* code size or even the binary file size.

Contents

Examples

The following JavaScript algorithm has a large number of redundant variables, unnecessary logic and inefficient string concatenation.

```
// Complex
function TK2getImageHTML(size, zoom, sensor, markers) {
    var strFinalImage = "";
    var strHTMLStart = '<img src="';
    var strHTMLEnd = '" alt="The map"/>';
    var strURL = "http://maps.google.com/maps/api/staticmap?center=";
    var strSize = '&size='+ size;
    var strZoom = '&zoom='+ zoom;
    var strSensor = '&sensor='+ sensor;

    strURL += markers[0].latitude;
    strURL += ",";
    strURL += markers[0].longitude;
    strURL += strSize;
    strURL += strZoom;
    strURL += strSensor;

    for (var i = 0; i < markers.length; i++) {
        strURL += markers[i].addMarker();
    }

    strFinalImage = strHTMLStart + strURL + strHTMLEnd;
    return strFinalImage;
};
```

The same logic can be stated more efficiently as follows:

```
// Simplified
const TK2getImageHTML = (size, zoom, sensor, markers) => {
    const [ { latitude, longitude } ] = markers;
    let url = `http://maps.google.com/maps/api/staticmap?center=${ latitude },${ longitude
}&size=${ size }&zoom=${ zoom }&sensor=${ sensor }`;

    markers.forEach(marker => url += marker.addMarker());
```

```
    return `<img src="${ url }" alt="The map" />`;
};
```

Code density of different languages

The difference in code density between various computer languages is so great that often less memory is needed to hold both a program written in a "compact" language (such as a domain-specific programming language, Microsoft P-Code, or threaded code), plus an interpreter for that compact language (written in native code), than to hold that program written directly in native code.

Reducing bloat

Some techniques for reducing code bloat include:[1]

- Code refactoring commonly used code sequence into a subroutine, and calling that subroutine from several locations, rather than copy and pasting the code at each of those locations.
- Re-using subroutines that have already been written (perhaps with additional parameters), rather than re-writing them again from scratch as a new routine.

See also

- Dead code elimination
- Minimalism (computing)
- Muntzing
- Polymorphism (computer science)
- Software optimization
- Software bloat
- Lightweight software

References

1. "Code bloat" (http://docforge.com/wiki/Code_bloat). *DocForge*. Retrieved 30 December 2009.