Program and Resource Caching Overview

August 2006

By

Jon Bradley – Software Engineer, BASIS International Ltd.

This tutorial introduces developers to number of cache mechanisms that improve the performance of loading BBj program and resource files.

Contents

Program and Resource Caching Overview

Introduction

In general, caching reduces load times by retaining a copy of a file in memory. BBj provides a number of cache mechanisms that improve the performance of loading BBj program and resource files.

Abstract

Loading programs or resources follow a number of steps. Some of these steps may involve accessing the hard drive; a slow operation even on modern computers. Additional processing of files, such as compilation, further slows the process of loading programs and resources. By storing recently used programs or resources in cache, BBj can minimize load times. There are ways in which a programmer or system administrator can affect the amount of time required to load programs or resources. **PREFIX, ADDR,** memory settings, and *pinning* have a significant effect on performance.

PREFIX
The prefix is a list of directories an interpreter will search for a file. The prefix begins implicitly with the current working directory. The **PREFIX** verb configures the search path at runtime. More commonly, the config file configures the search prefix. Both the order and length of the prefix can have a significant effect on performance. See **PREFIX** and **Configuration Files** for more information.

ADDR
ADDR is an interpreter verb, used with in a BBj program to insert a program into an interpreter local cache for the lifespan of that interpreter. A program that has been **ADDR**'d will never change or drop from cache. Use the **DROP** verb to remove the entry. See **ADDR** and **DROP** for more information on using ADDR.

Memory Settings of BBjServices

The amount of memory allotted for BBjServices determines how many programs and resources BBjServices attempts to cache. Read about memory configuration in the following article published in the *BASIS International Advantage*: **New Language Features Give Programmers More Choices** and in the online documentation at **Configuring BBj Services via the Enterprise Manager**.

Program and Resource Pinning

Program and resource pinning determines whether BBj checks the disk to ensure it is loading the current, most eligible file within the **PREFIX.** This is a potentially expensive operation, which may not be necessary in a production environment where the programs and resources are not changing. To improve load time performance considerably, avoid performing the prefix scan and up-to-date check by choosing ***production*** for this setting in **Enterprise Manager**.

Since a program or resource may 'fall out of cache' at any time, BBj may load the most current version at any time. See the **Configuring BBj Services via the Enterprise Manager** for more information.

The Resolved Name Map

Each interpreter maintains a local mapping of relative file names to their resolved canonical file names. The interpreter, when *pinning* is set to ***production,*** uses this map in place of scanning the **PREFIX**, when possible. The **PREFIX, SETDRIVE,** and **CHDIR** verbs clear the resolved name map.

Loading a Program or Resource

A program is loaded and placed in cache via **LOAD**, **CALL**, **RUN, START, SCALL,** or **`BBjAPI.newBBjSession()`**. A resource is loaded and placed in cache via **'RESOPEN' or `BBjSysGui.resOpen()`**.

Assume the file specified to load is **`foo.src`.** This loading process follows these steps:

The interpreter local **ADDR** cache is checked for a program which has been **ADDR**'d with exactly `foo.src`. If such a program has been **ADDR**'d, that program is returned.

If pinning is set to ***production***, BBj checks the *resolved name map* to see if `foo.src` has already been resolved within that interpreter's **PREFIX** to a resolved canonical name in the server wide cache. If such a file exists in the server wide cache, it is returned.

Scan the interpreter's **PREFIX** for a file named `foo.src`. If the BBj filesystem finds a file within the prefix, BBj adds the resolved canonical name to the *resolved name map* and checks the server wide cache with this resolved canonical name.

If the program exists in cache and pinning is set to ***production***, this program is returned.

If the program exists in cache and pinning is set to ***development***, the 'last loaded' time of the program is checked against the **on disk** date of the file. If the **on disk** modified time is after the 'last loaded' time, BBj reloads from disk, caches and returns the program. If the program is not in cache, BBj loads, caches and returns the program.

Preload of Files at BBjServices Start

BBjServices writes out a list of its current cache when shutting down. By default, BBjServices preloads this list of files to cache upon startup. See **Configuring BBj Services via the Enterprise Manager**.in the online documentation for more information.

Files Dropping from Cache

BBjServices generally stores all programs and resources within cache. If BBjServices detects that memory is getting scarce, it will drop entries from the cache to make more room for the

Interpreters; most likely, the least recently used programs. When BBjServices drops a program from cache, a message appears in the log. If a great number of programs are frequently dropping from cache, increasing the memory settings of BBjServices may improve performance. To prevent the accumulation of temporary programs in cache BBjServices will periodically remove unused programs from the cache, regardless of memory scarcity.