Base Language - Feature #12008

implement Progress.IO.FileOutputStream

10/15/2026 07:29 PM - Greg Shah

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**reviewer:**	
vendor_id:	GCD	**production:**	No
case_num:		**env_name:**	
version_reported:		**topics:**	
version_resolved:			

Description

Related issues:

Related to Base Language - Feature #9488: implement all built-in OO classes u...	**WIP**
Related to Base Language - Feature #4658: OO serialization support	**WIP**

History

#1 - 10/15/2026 07:30 PM - Greg Shah

Implement Progress.IO.FileOutputStream

Project: Base Language
Related: #9488 (built-in OO classes), #4658 (OO serialization)

Problem

Progress.IO.FileOutputStream is a native built-in class, not ADE 4GL source, so #9488 cannot convert it. FWD's com.goldencode.p2j.oo.io package has InputStream, FileInputStream, OutputStream, MemoryOutputStream and ByteOrder. It has no FileOutputStream, and no MemoryInputStream either.

The class is needed for two reasons:

- **ADE conversion (#9488).** The ADE logging framework writes through it in OpenEdge.Logging.Writer.NamedFileWriter, OpenEdge.Logging.Writer.FileLogWriterBuilder and OpenEdge.Logging.Filter.LogFilterBuilder. So does OpenEdge.DataAdmin.Util.DomainSystemExport. Those classes will not compile in FWD without it.
- **OO serialization (#4658).** It is the usual target stream for Progress.IO.BinarySerializer and Progress.IO.JsonSerializer when an object is serialized to a file.

API (OpenEdge 12.8 skeleton)

```
class Progress.IO.FileOutputStream inherits Progress.IO.OutputStream final:
  constructor public FileOutputStream (fileName as character).
  constructor public FileOutputStream (fileName as character, append as logical).
  define public property Append   as logical   no-undo public get. private set.
  define public property FileName as character no-undo public get. private set.
  method public override void  Close ().
  method public override void  Flush ().
  method public override int64 Write (source as longchar).
  method public override int64 Write (source as character).
  method public override int64 Write (source as memptr, offset as int64, length as int64).
end.
```

Work

1. Implement the class in com.goldencode.p2j.oo.io, using FileInputStream as the model for the structure and annotations. The file must be opened through FWD's file system layer, so that it is created where the 4GL session would create it: on the FWD client, relative to the session's working directory, with PROPATH not consulted.
2. Establish the OpenEdge behaviour and error numbers for each of the following, and match them:
   - opening a file that cannot be created;
   - writing after Close;
   - closing twice;
   - an offset or length outside the bounds of the memptr.
3. Establish which codepage Write(character) and Write(longchar) use. The candidates are -cpstream, -cpinternal and the longchar's own codepage.
4. Implement MemoryInputStream at the same time if it is small. It is the read-side counterpart of MemoryOutputStream, and #4658 will probably need it to deserialize from memory.
5. Annotate both classes with accurate LegacyResourceSupport levels.

Testcases

Write the tests against OpenEdge 12.8. Cover:

- both constructors, including append onto an existing file;
- each Write overload with ASCII and non-ASCII data, checking the bytes on disk;
- the Flush and Close behaviour, and the error cases listed above;
- a round trip of writing with FileOutputStream and reading back with FileInputStream.

#2 - 10/15/2026 07:44 PM - Greg Shah

- Related to Feature #9488: implement all built-in OO classes using the open source ADE code from 12.8.4 added

#3 - 10/15/2026 07:45 PM - Greg Shah

- Related to Feature #4658: OO serialization support added