Subject: Source files are moved to <temp-aux> during debugging under BSD
Posted by Mindtraveller on Thu, 26 Apr 2012 13:42:53 GMT
View Forum Message <> Reply to Message

I'm developing app under FreeBSD. While it is executed under debugger and stopped on breakpoint, this source file is considered part of <temp-aux>, not a part of current package.

This bug works for all versions of g++, and it appeared very long ago (at least 2-3 years).

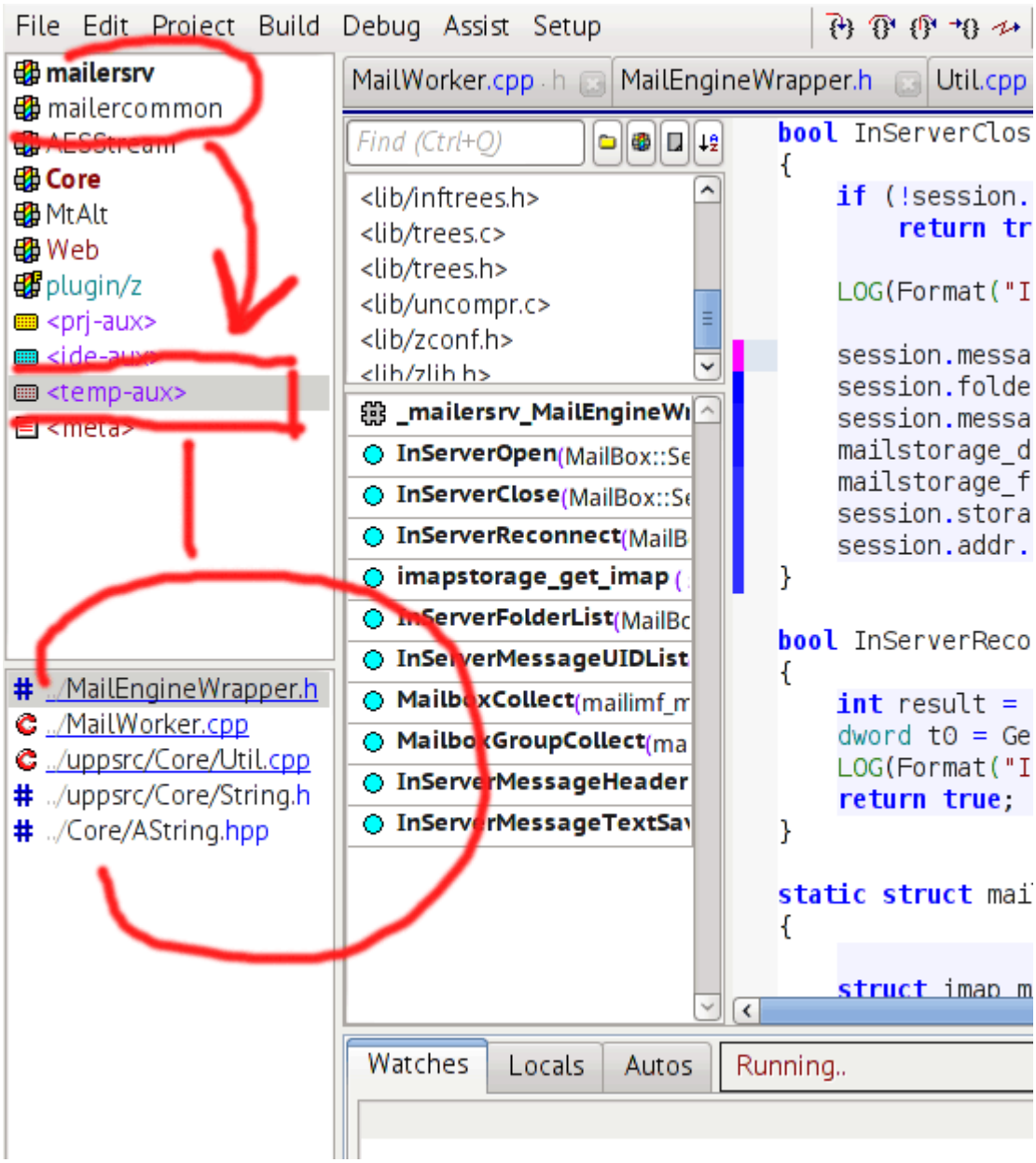
May be Linux developers have the same issue.

Actually I'd like to help eliminating this bug but TheIDE has much code and I would really appreciate any clues whare to start looking at.

Subject: Re: Source files are moved to <temp-aux> during debugging under BSD
Posted by Mindtraveller on Fri, 27 Apr 2012 09:07:34 GMT
View Forum Message <> Reply to Message

Attached screenshot to make situation more clear.

File Attachments
1) upp-temp-aux.png, downloaded 703 times



Subject: Re: Source files are moved to <temp-aux> during debugging under BSD
Posted by Mindtraveller on Mon, 07 May 2012 20:29:16 GMT
View Forum Message <> Reply to Message

Anyone, please?
I'd greatly appreciate any clues where in TheIDE sources to start finding this bug.
Thanks in forward.

Subject: Re: Source files are moved to <temp-aux> during debugging under BSD
Posted by Sender Ghost on Fri, 25 May 2012 21:23:15 GMT

Hello, Pavel.
Mindtraveller wrote on Thu, 26 April 2012 15:42I'm developing app under FreeBSD. While it is executed under debugger and stopped on breakpoint, this source file is considered part of <temp-aux>, not a part of current package.
Mindtraveller wrote on Mon, 07 May 2012 22:29I'd greatly appreciate any clues where in TheIDE sources to start finding this bug.
uppsrc/ide/ide.key(68) ->
Toggle Spoiler

```
KEY(DEBUG, "Run (in debugger)", K_F5)
```

uppsrc/ide/idebar.cpp(483) ->
Toggle Spoiler

```
menu.Add(b, AK_DEBUG, THISBACK1(BuildAndDebug, false))
 .Help("Build application & run debugger");
```

uppsrc/ide/Debug.cpp(162) ->
Toggle Spoiler

```
void Ide::BuildAndDebug(bool runto)
{
 VectorMap<String, String> bm = GetMethodVars(method);
 String builder = bm.Get("BUILDER", "");
 if(!Build())
  return;
 if(!FileExists(target))
  return;
 if(designer)
  EditAsText();
 One<Host> host = CreateHostRunDir();
 host->ChDir(Nvl(rundir, GetFileFolder(target)));
 HideBottom();
 editor.Disable();
#ifdef COMPILER_MSC
 if(builder == "GCC")
  if(gdbSelector)
   debugger = Gdb_MI2Create(host, target, runarg);
  else
   debugger = GdbCreate(host, target, runarg);
 else
  debugger = PdbCreate(host, target, runarg);
#else
 if(gdbSelector)
  debugger = Gdb_MI2Create(host, target, runarg);
```

```
 else
  debugger = GdbCreate(host, target, runarg);
```

uppsrc/ide/Debug.cpp(188) ->
uppsrc/ide/Debuggers/Gdb.cpp(598) ->
Toggle Spoiler

```
One<Debugger> GdbCreate(One<Host> host, const String& exefile, const String& cmdline)
{
 Gdb *dbg = new Gdb;
 if(!dbg->Create(host, exefile, cmdline)) {
  delete dbg;
  return NULL;
 }
 return dbg;
}
```

uppsrc/ide/Debuggers/Gdb.cpp(601) ->
uppsrc/ide/Debuggers/Gdb.cpp(554) ->
uppsrc/ide/Debuggers/Debuggers.h(126) ->
uppsrc/ide/Debuggers/Debuggers.h(140) ->
uppsrc/ide/Debuggers/Gdb.cpp(321) ->
Toggle Spoiler

```
void Gdb::Step(const char *cmd)
{
 bool b = disas.HasFocus();
 String s = Cmdp(cmd);
 if(b) disas.SetFocus();
 CheckEnd(s);
 IdeActivateBottom();
}
```

uppsrc/ide/Debuggers/Gdb.cpp(324) ->
uppsrc/ide/Debuggers/Gdb.cpp(262) ->
Toggle Spoiler

```
String Gdb::Cmdp(const char *cmdline, bool fr)
{
 String s = Cmd(cmdline);
 if(ParsePos(s, file, line, addr)) {
  IdeSetDebugPos(GetLocalPath(file), line - 1, fr ? DbgImg::FrameLinePtr()
   : DbgImg::IpLinePtr(), 0);
  IdeSetDebugPos(GetLocalPath(file), line - 1,
   disas.HasFocus() ? fr ? DbgImg::FrameLinePtr() : DbgImg::IpLinePtr()
```

```
   : Image(), 1);
  SyncDisas(fr);
  autoline = IdeGetLine(line - 2) + ' ' + IdeGetLine(line - 1) + ' ' + IdeGetLine(line);
 }
```

uppsrc/ide/Debuggers/Dbg.cpp(90)
uppsrc/ide/Debuggers/Gdb.cpp(228)
Toggle Spoiler

```
bool ParsePos(const String& s, String& fn, int& line, adr_t & adr)
{
 const char *q = FindTag(s, "\x1a\x1a");
 if(!q) return false;
 q += 2;
 Vector<String> p = Split(q + 2, ':');
 p.SetCount(5);
 fn = String(q, q + 2) + p[0];
```

The parsed path returns to fn variable above, which is file variable inside Gdb::Cmdp method.
uppsrc/ide/Core/Core.cpp(230) ->
uppsrc/ide/Core/Core.cpp(233) ->
uppsrc/ide/idewin.cpp(201) ->
Toggle Spoiler

```
void Ide::IdeSetDebugPos(const String& file, int line, const Image& img, int i)
{
 posfile[i] = file;
 posline[i] = line;
 posimg[i] = img;
 EditFile(file);
```

uppsrc/ide/idewin.cpp(206) ->
uppsrc/ide/idefile.cpp(554) ->
Toggle Spoiler

```
void Ide::EditFile(const String& p)
{
 if(p.IsEmpty()) {
  FlushFile();
  return;
 }
 if(p == HELPNAME) {
  if(designer && designer->GetFileName() == p)
   return;
  package.FindSetCursor(METAPACKAGE);
```

```
 filelist.FindSetCursor(HELPNAME);
 return;
}

String path = NormalizePath(p);
if(designer ? path == designer->GetFileName() : path == editfile)
 return;

FlushFile();
if(path.IsEmpty())
 return;

for(int i = 0; i < package.GetCount(); i++) {
 String pkg = package[i].name;
 Package p;
 p.Load(PackagePathA(pkg));
 for(int j = 0; j < p.file.GetCount(); j++)
  if(PathIsEqual(SourcePath(pkg, p.file[j]), path)) {
   package.FindSetCursor(pkg);
   ShowFile(j);
```

uppsrc/ide/idefile.cpp(583) ->
uppsrc/ide/UppWspc.cpp(792).
Toggle Spoiler

```
void  WorkspaceWork::ShowFile(int pi)
{
	bool open = true;
	Sepfo sf;
	for(int i = 0; i < actual.file.GetCount(); i++) {
		if(actual.file[i].separator) {
			sf = Sepfo(GetActivePackage(), actual.file[i]);
			open = closed.Find(sf) < 0;
		}
		else
		if(i == pi) {
			if(!open) {
				int i = filelist.GetCursor();
				int s = filelist.GetSbPos();
				closed.UnlinkKey(sf);
				SaveLoadPackage(false);
				filelist.SetSbPos(s);
				filelist.SetCursor(i);
			}
			return;
		}
	}
```

}

I could confirm this situation on FreeBSD and Linux operating systems, in case of package nests of assembly, configured with symlink paths. In your case "/home" is symlink to "/usr/home", therefore, when gdb returns "/usr/home/.." file path, TheIDE couldn't find it for "/home/.." file workspace paths and opens them inside <temp-aux> instead.
The quick solution here is to use absolute paths for package nests of assembly.
