Clang Vendor Options

Control new Clang and LLVM changes in a release

Alex Lorenz aleksei_lorenz@apple.com | LLVM Distributors Conf | September 16th 2021

Apple Clang Toolchain

- Apple Clang builds operating systems like iOS
- New Clang not allowed to break anything
- Typical adoption pattern: pick up new Clang twice a year
- Long-term goal: adopt new Clang more frequently

Keeping Up With Main

- Build and test Apple’s OSes weekly/daily
- LLVM & Clang get 100s of changes a week on main, causing
  - New compile-time errors
  - Crashes, test failures, or unexpected runtime issues
- Impacts the ability to test Clang continuously
- Complicates releases - issues accumulate and hide issues

Managing Changes

- Manage compiler changes which impact continuous testability
- Compiler flags and options disable or tweak changes
- Downstream vendor options and flags support

Vendor Options

- Add Clang language options declaratively downstream
- Control when the driver enables it
- Generic clang flag `-clang-vendor-feature=+<name>`
- Require some downstream Sema/CodeGen changes

```
// Disable 16d03818412415c56efcd482d18c0cbdf712524c
VENDOROPT(thisNoAlignAttr, 1, 0,
          "needed because of this issue: …",
          [](const llvm::Triple &TT) -> bool {
  return /* is enabled */ TT.isOSDarwin();
})
```

Vendor Options Example

```
// Disable 16d03818412415c56efcd482d18c0cbdf712524c
VENDOROPT(thisNoAlignAttr, 1, 0,
          “needed because of this issue: …”,
          [](const llvm::Triple &TT) -> bool {
  return /* is enabled */ TT.isOSDarwin();
})
```

```
diff --git a/clang/lib/CodeGen/CGCall.cpp b/clang/lib/CodeGen/CGCall.cpp
--- a/clang/lib/CodeGen/CGCall.cpp
+++ b/clang/lib/CodeGen/CGCall.cpp
@@ -2361 +2361,2 @@
-    Attrs.addAlignmentAttr(Alignment);
+    if (!getContext().getLangOpts().thisNoAlignAttr)
+      Attrs.addAlignmentAttr(Alignment);
```

Vendor Options Example

```
// Disable 16d03818412415c56efcd482d18c0cbdf712524c
VENDOROPT(thisNoAlignAttr, 1, 0,
          "needed because of this issue: …",
          [](const llvm::Triple &TT) -> bool {
  return /* is enabled */ TT.isOSDarwin();
})
```

Typically only minimal changes needed

```
diff --git a/clang/lib/CodeGen/CGCall.cpp b/clang/lib/CodeGen/CGCall.cpp
--- a/clang/lib/CodeGen/CGCall.cpp
+++ b/clang/lib/CodeGen/CGCall.cpp
@@ -2361 +2361,2 @@
-    Attrs.addAlignmentAttr(Alignment);
+    if (!getContext().getLangOpts().thisNoAlignAttr)
+      Attrs.addAlignmentAttr(Alignment);
```

Vendor Flags

- Add vendor specific `-cc1 / -mllvm` flags to Clang invocations
- Control when driver sets them
- Require implementation of flags if not yet implemented

```
VENDOR_LLVM_FLAG("-inline-deferral-scale=-1",
                 "Partially revert cec20db58825 because of this issue: …",
                 [](const llvm::Triple &TT) -> bool {
  return /* is enabled */ …;
})
```

Controlling New Warnings

- Disable new warnings that cause a lot of build failures
- Users can still pass `-W<name>` to re-enable them
- Disable `-Werror` promotion for specific user projects
- Re-enable them when user code has been fixed

```
DISABLE_WARNING("suggest-override",
                "needed because of these issues: …")

DISABLE_PROMOTED_ERROR_FOR_PROJECT("deprecated-copy",
                                   "WebKit", "needed because of this issue: …")
```

Clang Integration Benefits

- Compiler testing is not blocked by unresolved issues
- Minimal downstream diff for each change
- Controllable - apply for an OS / project only
- Automated report generation

Upstream Support

- We plan to upstream the vendor option and flag harness
- Vendors could add their own options or flags
- Reach out if interested

Conclusion

- Vendor options and flags control new Clang changes
- Downstream code diff is minimized
- Continuous testing of Apple’s OSes no longer blocked
- Downstream main branch better tested and qualified
- Release management simplified