Problem

Although the operations are very similar, the cache is not re-used by formatters in the lint goal. This means they are run twice when running `./pants fmt lint`, once for `fmt` and once for `lint`.

Additionally, the output of formatters in either goal provides minimal, yet varying, output. E.g. `docformatter` in `lint` simply outputs which files failed. `black` serves up emojis.

Solution

Unify `fmt` and `lint` to use a single process invocation

Plugin authors only provide the necessary plumbing to run the tool for `fmt`, which Pants will use to create the Process to be run. Once the sandbox has been formatted Pants will either copy the files back to the workspace for `fmt` or diff the workspace for `lint`.

For formatters running with both `fmt` and `lint`, I propose we no longer output std{out,err} for a tool, other than at -ldebug level for folks to debug what the underlying tool is doing. Pants will instead output filenames which changed for `fmt` and output the filenames for `lint` on files which the files differed.

This has several benefits:

- Since there's only one set of args, the process result is cached between both `fmt` and `lint`. `./pants fmt lint` now only does exactly what's necessary and not anything more.
- The output among all formatters is consistent for both `fmt` and `lint`, as well as helpful for `lint` (where it might not have been before)
- Formatting plugin authors now only have to write one rule (how to format) instead of 2, reducing boilerplate
- We can add formatters that don't have the equivalent of a `--check`.

UX Changes

The specific output is up for debate, but to demonstrate what the change looks like with various tools and output.

Note that we do not change the format of dedicated linters like Flake8. Their output is semantically important, so we must preserve it.

Fmt

Before:

```
14:08:16.06 [INFO] Completed: Format with Autoflake - autoflake made no
changes.
14:08:17.52 [WARN] Completed: Format with Black - black made changes.
reformatted src/python/pants/backend/python/dependency_inference/rules.py

All done! ✨ 🍰 ✨
1 file reformatted, 11 files left unchanged.


14:08:18.58 [INFO] Completed: Format with docformatter - docformatter made no
changes.
14:08:18.61 [INFO] Completed: Format with isort - isort made no changes.
```

After:

```
14:08:16.06 [INFO] Completed: Format with Autoflake - autoflake made no
changes.
14:08:17.52 [WARN] Completed: Format with Black - black made changes.
src/python/pants/backend/python/dependency_inference/rules.py

14:08:18.58 [INFO] Completed: Format with docformatter - docformatter made no
changes.
14:08:18.61 [INFO] Completed: Format with isort - isort made no changes.
```

Lint

Before:

```
15:52:34.03 [INFO] Completed: Lint with regex patterns - regex-lint succeeded.
15:52:35.61 [INFO] Completed: Lint with isort - isort succeeded.
15:52:35.61 [ERROR] Completed: Lint with docformatter - docformatter succeeded.
15:52:35.62 [INFO] Completed: Lint with autoflake - autoflake succeeded.
15:52:35.62 [INFO] Completed: Lint with Black - black failed (exit code 1).
would reformat src/python/pants/backend/python/dependency_inference/rules.py

Oh no! 💥 💔 💥
1 file would be reformatted.
```

After:

```
15:52:34.03 [INFO] Completed: Lint with regex patterns - regex-lint succeeded.
15:52:35.61 [INFO] Completed: Format with isort - isort succeeded.
15:52:35.61 [ERROR] Completed: Lint with docformatter - docformatter succeeded.
15:52:35.62 [INFO] Completed: Lint with autoflake - autoflake succeeded.
15:52:35.62 [INFO] Completed: Format with Black - black failed.
src/python/pants/backend/python/dependency_inference/rules.py
```

Considerations

Since this wouldn't disrupt users on upgrade, this is not subject to the deprecation policy. As such, we'll keep Pants simple by switching to the new way wholesale.

If a user specifies arguments for a formatter when running either `fmt` or `lint` (but not both), they will invalidate the cache for the other. I view this as behavior that is inherent in Pants (the cache would've been invalidated for the specific command regardless) so users shouldn't be surprised.

Some possible implementations might have the lint output log contain `fmt` rule messages (e.g. "Completed: Format with black - black succeeded" output during `lint`). I personally see no issue with this as `black` is a formatter after all, but others may find the output slightly confusing.

The `lint` goal is runnable today on a read-only filesystem. When implementing the `lint` diff behavior, we might also need to handle the case that formatters might not be able to persist changes on disk. See https://github.com/pantsbuild/pants/issues/14489#issuecomment-1062236336. If this happens, we can still report a Lint failure (since the formatter attempted to change a file on disk), however

the filenames wouldn't be included in the output. We could however, provide a sensible default output message.

Metrics

On my 64-core 3.2Ghz:

Pants repo **Before**:

```
Benchmark 1: ./pants --no-pantsd --no-local-cache fmt lint ::
  Time (mean ± σ):     27.503 s ±  1.171 s    [User: 125.690 s, System: 18.363 s]
  Range (min … max):   25.446 s … 28.378 s    5 runs

Benchmark 1: ./pants --no-pantsd --no-local-cache
--process-execution-local-parallelism=4 fmt lint ::
  Time (mean ± σ):     44.069 s ±  0.951 s    [User: 110.673 s, System: 14.473 s]
  Range (min … max):   42.753 s … 45.211 s    5 runs
```

Pants repo **After**:

```
Benchmark 1: ./pants --no-pantsd --no-local-cache fmt lint ::
  Time (mean ± σ):     26.659 s ±  1.792 s    [User: 90.700 s, System: 13.856 s]
  Range (min … max):   23.464 s … 27.700 s    5 runs

Benchmark 1: ./pants --no-pantsd --no-local-cache
--process-execution-local-parallelism=4 fmt lint ::
  Time (mean ± σ):     37.086 s ±  1.044 s    [User: 77.548 s, System: 10.384 s]
  Range (min … max):   35.778 s … 38.345 s    5 runs
```

(The overall time remained the same for 64-cores but the performance is measurable for 4-cores, I suspect due to my 64-core machine blasting through the work and because linters take much longer than formatters. Take note of the User and System time however.)