tfdbg.DumpingDebugWrapperSession

Class **DumpingDebugWrapperSession**

Defined in `tensorflow/python/debug/wrappers/dumping_wrapper.py`.

See the guide: TensorFlow Debugger > Session wrapper class and `SessionRunHook` implementations

Debug Session wrapper that dumps debug data to filesystem.

Properties

`graph`

`graph_def`

`run_call_count`

`sess_str`

`session`

Methods

`__init__`

```
__init__(
    sess,
    session_root,
    watch_fn=None,
    thread_name_filter=None,
    pass_through_operrors=None,
    log_usage=True
)
```

Constructor of DumpingDebugWrapperSession.

Args:

- `sess`: The TensorFlow `Session` object being wrapped.

- `session_root` : (**`str`**) Path to the session root directory. Must be a directory that does not exist or an empty directory. If the directory does not exist, it will be created by the debugger core during debug `tf.Session.run` calls. As the **`run()`** calls occur, subdirectories will be added to **`session_root`** . The subdirectories' names has the following pattern: run__ E.g., run_1480734393835964_ad4c953a85444900ae79fc1b652fb324
- `watch_fn` : (**`Callable`**) A Callable that can be used to define per-run debug ops and watched tensors. See the doc of **`NonInteractiveDebugWrapperSession.__init__()`** for details.
- `thread_name_filter` : Regular-expression white list for threads on which the wrapper session will be active. See doc of **`BaseDebugWrapperSession`** for more details.
- `pass_through_operrors` : If true, all captured OpErrors will be propagated. By default this captures all OpErrors.
- `log_usage` : (**`bool`**) whether the usage of this class is to be logged.

Raises:

- `ValueError` : If **`session_root`** is an existing and non-empty directory or if **`session_root`** is a file.

`__enter__`

```
__enter__()
```

`__exit__`

```
__exit__(
    exec_type,
    exec_value,
    exec_tb
)
```

`as_default`

```
as_default()
```

`close`

```
close()
```

`increment_run_call_count`

```
increment_run_call_count()
```

`invoke_node_stepper`

```
invoke_node_stepper(
    node_stepper,
    restore_variable_values_on_exit=True
)
```

See doc of BaseDebugWrapperSession.invoke_node_stepper.

list_devices

```
list_devices(
    *args,
    **kwargs
)
```

make_callable

```
make_callable(
    fetches,
    feed_list=None,
    accept_options=False
)
```

on_run_end

```
on_run_end(request)
```

See doc of BaseDebugWrapperSession.on_run_end.

on_run_start

```
on_run_start(request)
```

See doc of BaseDebugWrapperSession.on_run_start.

on_session_init

```
on_session_init(request)
```

See doc of BaseDebugWrapperSession.on_run_start.

partial_run

```
partial_run(
    handle,
    fetches,
    feed_dict=None
)
```

partial_run_setup

```
partial_run_setup(
    fetches,
    feeds=None
)
```

Sets up the feeds and fetches for partial runs in the session.

prepare_run_debug_urls

```
prepare_run_debug_urls(
    fetches,
    feed_dict
)
```

Implementation of abstrat method in superclass.

See doc of `NonInteractiveDebugWrapperSession.prepare_run_debug_urls()` for details. This implementation creates a run-specific subdirectory under self._session_root and stores information regarding run `fetches` and `feed_dict.keys()` in the subdirectory.

Args:

- `fetches`: Same as the `fetches` argument to `Session.run()`
- `feed_dict`: Same as the `feed_dict` argument to `Session.run()`

Returns:

- `debug_urls`: (`str` or `list` of `str`) file:// debug URLs to be used in this `Session.run()` call.

`reset`

```
reset(
    *args,
    **kwargs
)
```

`run`

```
run(
    fetches,
    feed_dict=None,
    options=None,
    run_metadata=None,
    callable_runner=None,
    callable_runner_args=None
)
```

Wrapper around Session.run() that inserts tensor watch options.

Args:

- `fetches`: Same as the `fetches` arg to regular `Session.run()`.
- `feed_dict`: Same as the `feed_dict` arg to regular `Session.run()`.
- `options`: Same as the `options` arg to regular `Session.run()`.
- `run_metadata`: Same as the `run_metadata` arg to regular `Session.run()`.
- `callable_runner`: A `callable` returned by `Session.make_callable()`. If not `None`, `fetches` and `feed_dict` must both be `None`.
- `callable_runner_args`: An optional list of arguments to `callable_runner`.

Returns:

Simply forwards the output of the wrapped `Session.run()` call.

Raises:

- `ValueError`: On invalid `OnRunStartAction` value. Or if `callable_runner` is not `None` and either or both of `fetches` and `feed_dict` is `None`.

should_stop

```
should_stop()
```
