tfdbg.LocalCLIDebugWrapperSession

Contents

Class **`LocalCLIDebugWrapperSession`**

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

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

Concrete subclass of BaseDebugWrapperSession implementing a local CLI.

This class has all the methods that a `session.Session` object has, in order to support debugging with minimal code changes. Invoking its `run()` method will launch the command-line interface (CLI) of tfdbg.

Properties

`graph`

`graph_def`

`run_call_count`

`sess_str`

`session`

Methods

`__init__`

```
__init__(
    sess,
    dump_root=None,
    log_usage=True,
    ui_type='curses',
    thread_name_filter=None
)
```

Constructor of LocalCLIDebugWrapperSession.

Args:

- `sess`: The TensorFlow `Session` object being wrapped.
- `dump_root`: (`str`) optional path to the dump 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 `run()` calls and removed afterwards. If `None`, the debug dumps will be at tfdbg_ under the system temp directory.
- `log_usage`: (`bool`) whether the usage of this class is to be logged.
- `ui_type`: (`str`) requested UI type. Currently supported: (curses | readline)
- `thread_name_filter`: Regular-expression white list for thread name. See the doc of `BaseDebugWrapperSession` for details.

Raises:

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

`__enter__`

```
__enter__()
```

`__exit__`

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

`add_tensor_filter`

```
add_tensor_filter(
    filter_name,
    tensor_filter
)
```

Add a tensor filter.

Args:

- `filter_name`: (`str`) name of the filter.
- `tensor_filter`: (`callable`) the filter callable. See the doc string of `DebugDumpDir.find()` for more details about its signature.

`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
)
```

Overrides method in base class to implement interactive node stepper.

Args:

* `node_stepper`: (**`stepper.NodeStepper`**) The underlying NodeStepper API object.
* `restore_variable_values_on_exit`: (**`bool`**) Whether any variables whose values have been altered during this node-stepper invocation should be restored to their old values when this invocation ends.

Returns:

The same return values as the **`Session.run()`** call on the same fetches as the NodeStepper.

`list_devices`

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

`make_callable`

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

`on_run_end`

```
on_run_end(request)
```

Overrides on-run-end callback.

Actions taken: 1) Load the debug dump. 2) Bring up the Analyzer CLI.

Args:

* `request`: An instance of OnSessionInitRequest.

Returns:

An instance of OnSessionInitResponse.

`on_run_start`

```
on_run_start(request)
```

Overrides on-run-start callback.

Invoke the CLI to let user choose what action to take: `run` / `invoke_stepper`.

Args:

- `request`: An instance of `OnRunStartRequest`.

Returns:

An instance of `OnRunStartResponse`.

`on_session_init`

```
on_session_init(request)
```

Overrides on-session-init callback.

Args:

- `request`: An instance of `OnSessionInitRequest`.

Returns:

An instance of `OnSessionInitResponse`.

`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.

`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()
```