Implicit Request Contexts

On this page:

Using Implicit Request Contexts

In addition to explicit and per-proxy request contexts, you can also establish an implicit context on a communicator. This implicit context is sent with all invocations made via proxies created by that communicator, provided that you do not supply an explicit context with the call.

Access to this implicit context is provided by the `Communicator` interface:

Slice

```
module Ice {
    local interface Communicator
    {
        ImplicitContext getImplicitContext();

        // ...
    };
};
```

`getImplicitContext` returns the implicit context object. If a communicator has no implicit context, the operation returns null.

You can manipulate the contents of the implicit context via the `ImplicitContext` interface:

Slice

```
local interface ImplicitContext
{
    Context getContext();
    void    setContext(Context newContext);

    string get(string key);
    string put(string key, string value);
    string remove(string key);
    bool   containsKey(string key);
};
```

The `getContext` operation returns the currently-set context dictionary. The `setContext` operation replaces the currently-set context in its entirety.

The remaining operations allow you to manipulate specific entries:

- `get`
  This operation returns the value associated with `key`. If `key` was not previously set, the operation returns the empty string.
- `put`
  This operation adds the key-value pair specified by `key` and `value`. It returns the previous value associated with `key`; if no value was previously associated with `key`, it returns the empty string. It is legal to add the empty string as a value.
- `remove`
  This operation removes the key-value pair specified by `key`. It returns the previously-set value (or the empty string if `key` was not previously set).
- `containsKey`
  This operation returns true if `key` is currently set and false, otherwise. You can use this operation to distinguish between a key-value pair that was explicitly added with an empty string as a value, and a key-value pair that was never added at all.

Scope of the Implicit Context

You establish the implicit context on a communicator by setting a property, `Ice.ImplicitContext`. This property controls whether a communicator has an implicit context and, if so, at what scope the context applies. The property can be set to the following values:

- `None`
  With this setting (or if `Ice.ImplicitContext` is not set at all), the communicator has no implicit context, and `getImplicitContext` returns null.

- `Shared`
  The communicator has a single implicit context that is shared by all threads. Access to the context via its `ImplicitContext` interface is interlocked, so different threads can concurrently manipulate the context without risking data corruption or reading stale values.

- `PerThread`
  The communicator maintains a separate implicit context for each thread. This allows you to propagate contexts that depend on the sending thread (for example, to send per-thread transaction IDs).

See Also

- Explicit Request Contexts
- Per-Proxy Request Contexts