.NET async - deadlock and `ConfigureAwait`
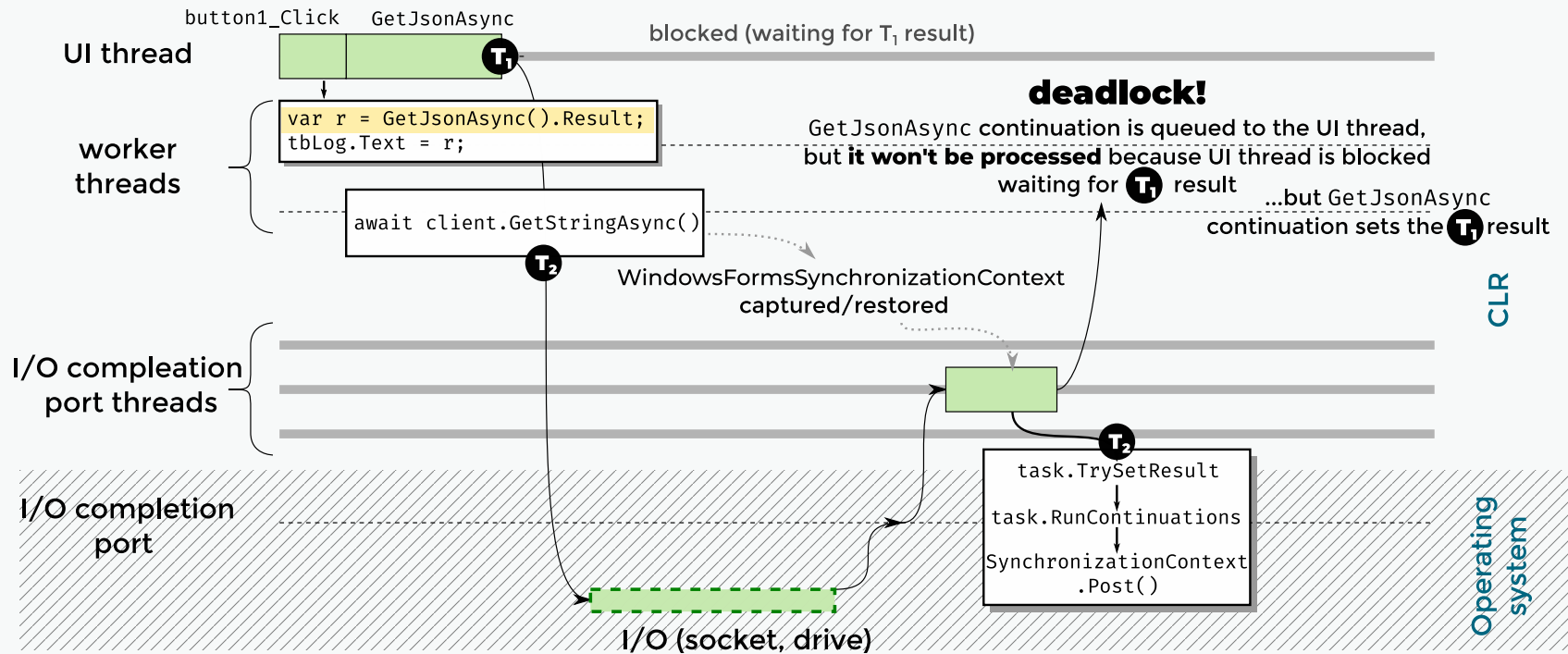
**.Result()
causing deadlock**

Underlying `Task` captures synchronization context

button1_Click   GetJsonAsync

UI thread

blocked (waiting for T_1 result)

deadlock!

GetJsonAsync continuation is queued to the UI thread, but **it won't be processed** because UI thread is blocked waiting for T_1 result

...but GetJsonAsync continuation sets the T_1 result

worker threads

```
var r = GetJsonAsync().Result;
tbLog.Text = r;
```

```
await client.GetStringAsync()
```

T_2

WindowsFormsSynchronizationContext captured/restored

CLR

I/O compleation port threads

I/O completion port

T_2

```
task.TrySetResult

task.RunContinuations

SynchronizationContext
.Post()
```

I/O (socket, drive)

Operating system

**.Result()
not causing deadlock**

Underlying `Task` does not capture synchronization context because of `ConfigureAwait(false)`

button1_Click   GetJsonAsync

UI thread

blocked (waiting for T_1 result)

button1_Click()

set completed (unblock)

worker threads

```
var r = GetJsonAsync().Result;
tbLog.Text = r;
```

```
await client.GetStringAsync()
          .ConfigureAwait(false);
```

T_2

thread pool scheduling used

GetJsonAsync

T_1

```
task.TrySetResult
```

no continuations to run (T_1 is not awaited, no state machine)

I/O compleation port threads

I/O completion port

T_2

```
task.TrySetResult

task.RunContinuations

SynchronizationContext
.Post()
```

I/O (socket, drive)

Operating system

AsyncExpert.com
@konradkokosa