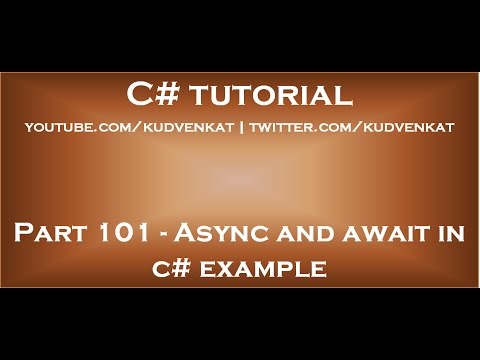
**Await Async**

<https://www.youtube.com/watch?v=C5VhaxQWcpE>

[](https://www.youtube.com/watch?v=C5VhaxQWcpE)

Asynchronne muzeme zavolat delegata Action. Mejme metodu:

protected override async void OnInitialize()

{

base.OnInitialize();

//**Kod pred Action probehne synchronne**

Action initAction = () =>

{

//**Kod uvnitr delegata probehne asynchronne**

};

await Task.Run(initAction); //**spusteni asynchronni operace v NOVEM VLAKNE**

//**Dalsi kod probehne synchronne, na kod uvnitr delegata se neceka.**

if (m\_partInfo != null)

{

// **V nasledujici (synchronni) casti je volana asynchronni metoda GetPartFromDatabase. Kod za volanim jede dal.**

var datacontract = **await GetPartFromDatabase**(PublicationWrapper.Data, m\_partInfo);

SelectedOrientation = Orientations.First(d => d.Id == Part.ScanningPreset.Orientation);

PartNameSelectedItem = PartNameComboItems.First(d => d.Id == Part.PartNameContract.Id);

}

}

// pokud je pouzite klicove slovo async predpoklada se ze uvnitr metody je alespon jenou pouzite await. Jinak metoda probiha synchronne. Tato metoda nevraci zadnou hodnotu, vraci **BEZICI ULOHU**

private async Task<PartDataContract> GetPartFromDatabase(PublicationDataContract publication, PartBasicInfoDataContract part)

{

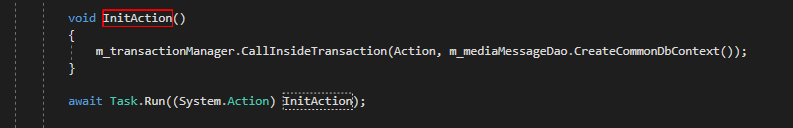
var response = await Task.Run(() => ClientToServicePublisher.Publish(new GetPartByIdRequest(part.Id)));

var result = response.contract;

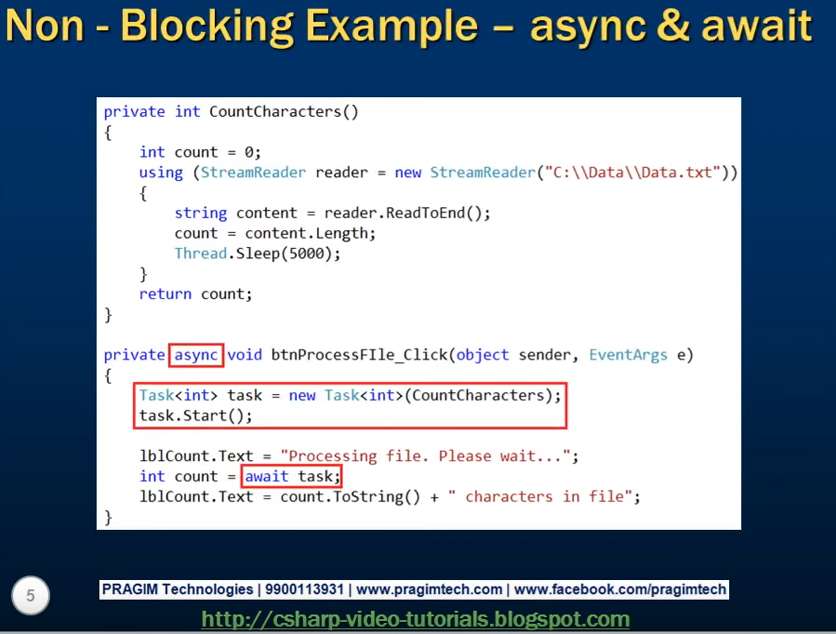
return result;

}

**Resharper to jeste upravi na :**



Obrazek z videa:



To, co je na obrazku je pouzite v Importovadle2 ImportGridViewModel

private async void DeleteImport()

{

if (MessageBox.Show(Localisation.DeleteSelectedImports) == MessageBoxResult.OK)

{

m\_informationService?.InvokeInformationOccured(this, Localisation.DeletingImports, InformationSeverity.Info);

m\_eventAggregator.Publish(new IsBusyWrapper(true));

var task = new Task<bool>(DoWork);

task.Start();

var result = await task;

if (result)

{

var importIds = m\_selectedProgressLines.Select(d => d.ImportId).ToList();

RemoveDeletedLines(importIds);

m\_informationService?.InvokeInformationOccured(this, Localisation.ImportsWereDeleted, InformationSeverity.Info);

}

else

{

m\_informationService?.InvokeInformationOccured(this, Localisation.FailedToDeleteImports, InformationSeverity.Error);

}

}

}

private bool DoWork()

{

var selectedLines = m\_selectedProgressLines;

var importIds = selectedLines.Select(d => d.ImportId).ToList();

try

{

void Action()

{

m\_tvImportItemDao.DeleteImportItemsByImportId(importIds);

m\_importDao.DeleteImportsByIds(importIds);

}

m\_transactionManager.CallInsideTransaction(Action, m\_importDao.CreateCommonDbContext());

}

catch (Exception ex)

{

if (s\_log.IsErrorEnabled)

s\_log.Error(ex);

return false;

}

finally

{

m\_eventAggregator.Publish(new IsBusyWrapper(false));

}

return true;

}

private void RemoveDeletedLines(List<int> importIds)

{

var linesToRemove = m\_selectedProgressLines.Where(d => importIds.Contains(d.ImportId)).ToList();

linesToRemove.ForEach(d => m\_progressLines.Remove(d));

NotifyOfPropertyChange(nameof(ProgressLines));

}