Блоки в сущности

Если в сущности есть блоки, то их можно мигрировать в ELMA365. Для этого нужно реализовать следующий код в точках расширения:

Для реализации миграции блока необходимо определить его в коде, например:

```
using System;
using System.Collections.Generic;
using ITino.ELMA.E365.Common.Models;
using Newtonsoft.Json;

namespace E365
{
    public class TestTablePartsDataItem : BaseDataItem
    {
        public override string Path => "app/_clients/testtableparts";

        [JsonProperty(PropertyName = "__name")]
        public string Name { get; set; }

        [JsonIgnore]
        public TableDataItem TestTable { get; set;}

        public TestTablePartsDataItem()
        {
            TestTable = new TableDataItem("testtable", this);
        }
    }
}
```

Структура блока и его записи заполняются для передачи, например (при этом указываются простые свойства и справочники):

```
using System;
using System.Collections.Generic;
using System.Linq;
using EleWise.ELMA.API;
using System.Text;
```

```
using EleWise.ELMA.ComponentModel;
using EleWise.ELMA.Runtime.NH.Listeners;
using NHibernate.Event;
using EleWise.ELMA.ConfigurationModel;
using EleWise.ELMA.Model.Common;
using ITino.ELMA.E365.Common.Components;
using EleWise.ELMA.Model.Entities;
using EleWise.ELMA.Extensions;
using ITino.ELMA.E365.Common.Managers;
using EleWise.ELMA;
using ITino.ELMA.E365.Common.Models;
using ITino.ELMA.E365.CRM.Listeners;
using ITino.ELMA.E365.Common.Listeners;

namespace E365
{
    [Component]
    public class TestTableParts_TestTableListener : PostFlushEventListener
    {
        public override void OnPostInsert(PostInsertEvent @event)
        {
            SyncItem(@event.Entity as ITestTableParts_TestTable, true);
        }

        public override void OnPostUpdate(PostUpdateEvent @event)
        {
            SyncItem(@event.Entity as ITestTableParts_TestTable, true);
        }

        public override void OnPostDelete(PostDeleteEvent @event)
        {
            SyncItem(@event.Entity as ITestTableParts_TestTable, true);
        }

        public static Pair<IBaseDataItem, IEntity> SyncItem(ITestTableParts_TestTable item,
bool syncLink = false)
        {
            return TestTablePartsListener.SyncItem(item?.Parent);
        }
    }
```

```
    [Component]
    public class TestTablePartsListener : PostFlushEventListener
    {
        public override void OnPostInsert(PostInsertEvent @event)
        {
            SyncItem(@event.Entity as ITestTableParts, true);
        }

        public override void OnPostUpdate(PostUpdateEvent @event)
        {
            SyncItem(@event.Entity as ITestTableParts, true);
        }

        public static Pair<IBaseDataItem, IEntity> SyncItem(ITestTableParts item, bool
syncLink = false)
        {
            var data = new TestTablePartsDataItem();

            if (item == null)
                return new Pair<IBaseDataItem, IEntity>(data, item);

            data.Uid = item.Uid;
            data.Name = item.Name;

            item.TestTable?.ForEach(tableItem => {
                var row = new RowDataItem();
                row.SimpleContext.Add ("boolean", tableItem.Boolean);
                row.SimpleContext.Add ("string", tableItem.String);
□□□□row.LinkedContext.Add ("currency", new List<LinkImpl> {
□□□□□new LinkImpl (CurrencyListener.SyncItem(tableItem.Currency))
□□□□});
□□□□row.LinkedContext.Add ("newuser", new List<LinkImpl> {
□□□□□new LinkImpl (UserListener.SyncItem(tableItem.NewUser))
□□□□});
                row.SimpleContext.Add ("newnumber", tableItem.NewNumber);
                row.SimpleContext.Add ("date", tableItem.Data);
                data.TestTable.Rows.Add(row);
            });

            E365DataItemManager.Instance.PushItem(typeof(TestTablePartsDataItem), item.Uid,
data, item, SR.T("TestTableParts: {0}", item.Name));
```

```
            return new Pair<IBaseDataItem, IEntity>(data, item);
        }
    }
}
```

Версия #2
Вербовик Максим Арнольдович создал 2023-02-21 09:02:19 MSK
Вербовик Максим Арнольдович обновил 2023-04-25 14:23:08 MSK