dev, Инвентаризация

Инвентаризация: model: fields

Описание атрибутов модели делаем в одной функции:

```
    public $parentAttr='parentService'; //имя атрибута в котором у нас объект-предок (родительское помещение/сервис/отдел и т.д.)
    public function attributeData() {
        return [
            ['model_id'] => [
                //позже транслируется в attributeLabels, используется в отображении атрибута в формах
                'label'=>'Модель ПК',
                //позже транслируется в attributeHints, используется для пояснения атрибута в формах
                'hint'=>'Модель системного блока / ноутбука. '
                        .'Если нужная модель отсутствует в списке, то нужно сначала завести в ее в соотв. категории оборудования',
                //позже транслируется в attributeIndexLabels, используется в отображении атрибута в списках
                'indexLabel'=>'Модель ПК',
                //позже транслируется в attributeIndexHints, используется для пояснения атрибута в списках
                'indexHint'=>'Модели можно перечислить варианты через верт. черту &quot;|&quot;',
                //чем заполнить селектор в форме, если значение не введено
                'placeholder' => 'Модель ПК не выбрана',
                //поле наследуемое (если не задано в этом объекте, то значение берется из родителя)
                'is_inheritable'=>true,
                //поле является ссылкой на объекты с обратной ссылкой на этот.
                //при наличии объектов в этом поле, считаем что себя удалять нельзя,
                //иначе в объектах ссылающихся на нас будут битые ссылки
                //(если не объявлять явно, определяется из $linkSchema
                'is_reverseLink'=false,
                //при вызове функции absorb это поле нужно поглощать из переданного объекта
                // - false - нет (по умолчанию для полей - "не обратных ссылок"
                // - 'ifemty' - если локальное значение отсутствует, то принимать с абсорбируемого объекта
                // - true - да (по умолчанию для полей, являющихся обратными ссылками)
                'absorb'=>true,
            ],
            'model'=>['alias'=>'model_id'], //значения параметров атрибута брать из model_id
        ]
    }
```

```
    /**
     * @return array[] Ссылками на объекты каких классов являются атрибуты
     * $linksSchema=[
     *      'services_ids'=>[
     *          Service::class,         //на какой класс ссылаемся
     *          'acls_ids',             //если там есть обратная ссылка, то в каком
аттрибуте
     *      ],
     *      'user_id'=>[
     *          Users::class,
     *          'loader'=>'user'     //как загрузчик этого объекта называется в
мастер-классе
     *      ],
     * ];
     */
    public $linksSchema=[
        'depends_ids' =>                [Services::class,'dependants_ids'],
        'comps_ids' =>                  [Comps::class,'services_ids'],
        'techs_ids' =>                  [Techs::class,'services_ids'],
        'maintenance_reqs_ids'=>
[MaintenanceReqs::class,'services_ids'],
        'maintenance_jobs_ids'=>
[MaintenanceJobs::class,'services_ids'],
        'support_ids' =>
[Users::class,'support_services_ids','loader'=>'support'],
        'infrastructure_support_ids' =>
[Users::class,'infrastructure_support_services_ids','loader'=>'infrastructur
eSupport'],
        'contracts_ids' =>              [Contracts::class,'services_ids'],
        'acls_ids' =>                   [Acls::class,'services_id'],

        'responsible_id' =>             [Users::class,'services_ids'],
        'infrastructure_user_id' =>
[Users::class,'infrastructure_services_ids','loader'=>'infrastructureRespons
ible'],
        'providing_schedule_id' =>
[Schedules::class,'providing_services_ids'],
        'support_schedule_id' =>
[Schedules::class,'support_services_ids'],
        'segment_id' =>                   [Segments::class,'services_ids'],
        'parent_id' =>
[Services::class,'children_ids','loader'=>'parentService'],
        'partners_id' =>                [Partners::class,'services_ids'],
        'places_id' =>                  [Places::class,'services_ids'],
        'currency_id' =>                Currency::class,
    ];

    /**
     * @inheritdoc
```

```
     */
    public function attributeLabels()
    {
        $labels=[];
        foreach ($this->attributeData() as $key=>$data)
            if (isset($data['label']))
                $labels[$key]=$data['label'];
        return $labels;
    }

    /**
     * @inheritdoc
     */
    public function attributeHints()
    {
        $hints=[];
        foreach ($this->attributeData() as $key=>$data)
            if (isset($data['hint']))
                $hints[$key]=$data['hint'];
        return $hints;
    }
```

From:
http://wiki.reviakin.net/ - **Wiki**

Permanent link:
http://wiki.reviakin.net/%D0%B8%D0%BD%D0%B2%D0%B5%D0%BD%D1%82%D0%B0%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F:dev:model:fields?rev=1716746105

Last update: **2024/05/26 17:55**