UpdateLoadProvider

boolean

true: onLoadtrueload

false

UpdateLoadProvider

```
/**
 *
 * boolean
 * true: onLoadtrueload
 * false
 *
 * @author James.Zhao
 * @version 5.1.3
 * Created by James.Zhao on 2020/4/8
 */
public interface UpdateLoadProvider extends Mutable {

    String XML_TAG = "UpdateLoadProvider";

    int CURRENT_LEVEL = 1;

    /**
     *
     */
    <T extends AbstractLoadProperty> boolean onStart(TaskInfoBO bo, AbstractLoad<T> load);

    /**
     *
     */
    <T extends AbstractLoadProperty> boolean onLoad(TaskInfoBO bo, AbstractLoad<T> load);

    /**
     *
     */
    <T extends AbstractLoadProperty> boolean onCommit(TaskInfoBO bo, AbstractLoad<T> load);

    /**
     *
     */
    <T extends AbstractLoadProperty> boolean onRename(TaskInfoBO bo, AbstractLoad<T> load);

    /**
     *
     */
    <T extends AbstractLoadProperty> boolean onEnd(TaskInfoBO bo, AbstractLoad<T> load);

    /**
     *
     */
    <T extends AbstractLoadProperty> boolean onRollback(TaskInfoBO bo, AbstractLoad<T> load, Throwable
throwable);
}
```

```
<extra-core>
        <UpdateLoadProvider class=""/>
</extra-core>
```

AbstractWidgetQueryAspectProvider

```
/**
 *
 *
 * @author James.Zhao
 * @version 5.1.3
 * Created by James.Zhao on 2020/4/8
 */
@API(level = UpdateLoadProvider.CURRENT_LEVEL)
public abstract class AbstractUpdateLoadProvider implements UpdateLoadProvider {
    @Override
    public int currentAPILevel() {
        return CURRENT_LEVEL;
    }

    @Override
    public String mark4Provider() {
        return getClass().getName();
    }

    @Override
    public <T extends AbstractLoadProperty> boolean onLoad(TaskInfoBO bo, AbstractLoad<T> load) {
        return false;
    }

    @Override
    public <T extends AbstractLoadProperty> boolean onCommit(TaskInfoBO bo, AbstractLoad<T> load) {
        return false;
    }

    @Override
    public <T extends AbstractLoadProperty> boolean onRename(TaskInfoBO bo, AbstractLoad<T> load) {
        return false;
    }

    @Override
    public <T extends AbstractLoadProperty> boolean onEnd(TaskInfoBO bo, AbstractLoad<T> load) {
        return false;
    }

    @Override
    public <T extends AbstractLoadProperty> boolean onRollback(TaskInfoBO bo, AbstractLoad<T> load, Throwable
throwable) {
        return false;
    }

    @Override
    public <T extends AbstractLoadProperty> boolean onStart(TaskInfoBO bo, AbstractLoad<T> load) {
        return false;
    }
}
```