

API komend workflow

Szablon komendy

```

<?php
require_once(COMMANDS_DIR.'AbsCommand.inc');
require_once(LIB_PATH.'util/Strings.inc');

/**
 *
 *
 * @uses AbsCommand
 * @uses ICommand
 * @final
 */
final class DeletePropertiesCommand extends AbsCommand implements ICommand {

    /**
     * getDescription
     *
     * @static
     * @access public
     * @return void
     */
    public static function getDescription() {

        return Translator::translate('Usuń przypisania parametrów procedury');

    }

    /**
     * getExpandedDescription
     *
     * @param string $params
     * @static
     * @access public
     * @return void
     */
    public static function getExpandedDescription($params = NULL) {

        return Translator::translate('Komenda służy do usunięcia parametrów procedury');

    }

    /**
     * getCommandApi
     *
     * @param string $params
     * @static
     * @access public
     * @return void
     */
    public static function getCommandApi($params = NULL) {

        $api = array(

```

```

    );

    return $api;
}

/**
 * execute
 *
 * @param Bean $bean
 * @param string $params
 * @access public
 * @return void
 */
public function execute(Bea $bean, $params) {

    if (!$bean instanceof Document) {
        $this->setMessage(Translator::translate('Komenda może być wykonana jedynie dla dokumentu.'), 'WARNING');
        $this->setMessage(Translator::translate('Komenda nie została wykonana.'), 'ERROR');
        throw new CommandException($this);
    }

    $data['doc_id'] = $bean->get('doc_id');
    $test = $this->db->delete('bpm_property_values', 'id_____ = 1 AND procid=(
        SELECT procid FROM documents WHERE doc_id = '.$data['doc_id'].' LIMIT 1)');

    $this->setMessage(Translator::translate('Komenda została wykonana poprawnie.'.$test), 'SUCCESS');

    return TRUE;
}
} // class RegisterDocumentCommand
?>

```