Implementing a new Editor Type in Bforartists / Blender 2.76

Table of content

Preface

This document describes the necessary steps to implement a new editor in Blender 2.76 / Bforartists. It's how we have implemented it in Bforartists. And it is not necessarily how it should be implemented. This document is not a tutorial, but a documentation of the steps that were necessary to implement a new editor in the Blender fork Bforartists. So don't expect deeper explanations.

Most code was simply copied from other places and reused then. This for example means that the one or another step and change may be in the wrong category. Or not even necessary. There may also be missing parts. The implementation was never completed.

For the first chapter, implementing the basic editor, we went through the official Blender tutorial. But this tutorial just covers the implementation of a naked editor. And it was from Blender 2.65. Already outdated with Blender 2.76, which is the base for Bforartists. So there was already lots of guessing and searching involved in the very first steps.

There are two Bforartists branches for this issue here, where you can compare the code too. The first branch is called Tutorial Editor. And can be found here:

https://github.com/Bforartists/Bforartists/tree/tutorialeditor

This part works perfectly fine. And it goes up to Theming entries first steps in this document.

The second part, called tutorialeditor-part2 is not so fine. It is mainly based at some loose files from somebody else. I got the theming working with that help. But was not able to implement / fix the content part. So be careful with the chapter Theming entries second part. It may or may not be valid. USE AT OWN RISK!

https://github.com/Bforartists/Bforartists/tree/tutorialeditor-part2

Reiner

03-07-2016

Adding a new editor

Note

This chapter here is taken from the official Blender tutorial that can be found here:
https://wiki.blender.org/index.php/Dev:Source/UI/Tutorials/AddAnEditor

It was once made for Blender 2.66, and then partially fixed for Blender 2.72. And leads you through the very basics. But this tutorial is with Blender 2.76 already quite outdated. The actual changes that are needed in Blender 2.76 differs from the changes mentioned in the tutorial. But it still works to get a basic editor type showing up.

The bad thing about this above tutorial is, it stops where it becomes interesting. How to get content like a menu into the editors. This will be covered in the later chapters behind the chapter Theming Entries first steps

Looking at the Source

Every space has an associated type defined by the *SpaceType* structure. The basic information contained within

the *SpaceType* structure is a name, an ID, a list of region types and pointers to a set of callback functions. The *SpaceType* structure is defined in source/blender/blenkernel/BKE_screen.h.

```
typedef struct SpaceType {
        struct SpaceType *next, *prev;

        char name[BKE_ST_MAXNAME];                  /* for menus */
        int spaceid;                                /* unique space identifier */
        int iconid;                                 /* icon lookup for menus */

        /* initial allocation, after this WM will call init() too */
        struct SpaceLink    *(*new)(const struct bContext *C);

        ...

        /* region type definitions */
        ListBase regiontypes;

        ...

} SpaceType;
```

A unique ID for each space type is defined in the *eSpace_Type* enumeration. When a new *SpaceType* instance is created it's *spaceid* is set to one of these values. *eSpace_Type* is defined in source/blender/makesdna/DNA_space_types.h.

```
/* space types, moved from DNA_screen_types.h */
/* Do NOT change order, append on end. types are hardcoded needed */
typedef enum eSpace_Type {
        SPACE_EMPTY    = 0,
        SPACE_VIEW3D   = 1,
        SPACE_IPO      = 2,
        SPACE_OUTLINER = 3,
        SPACE_BUTS     = 4,
        SPACE_FILE     = 5,
        SPACE_IMAGE    = 6,
        SPACE_INFO     = 7,
        SPACE_SEQ      = 8,
        SPACE_TEXT     = 9,
        SPACE_IMASEL   = 10, /* deprecated */
        SPACE_SOUND    = 11, /* Deprecated */
        SPACE_ACTION   = 12,
        SPACE_NLA      = 13,
        SPACE_SCRIPT   = 14, /* Deprecated */
        SPACE_TIME     = 15,
        SPACE_NODE     = 16,
        SPACE_LOGIC    = 17,
        SPACE_CONSOLE  = 18,
        SPACE_USERPREF = 19,
        SPACE_CLIP     = 20,

        SPACEICONMAX = SPACE_CLIP
} eSpace_Type;
```

When Blender's window manager is initialized it calls the *ED_spacetypes_init()* function in source/blender/editors/space_api/spacetypes.c which in turn calls each *ED_spacetype_*()* function to initialize each space type.

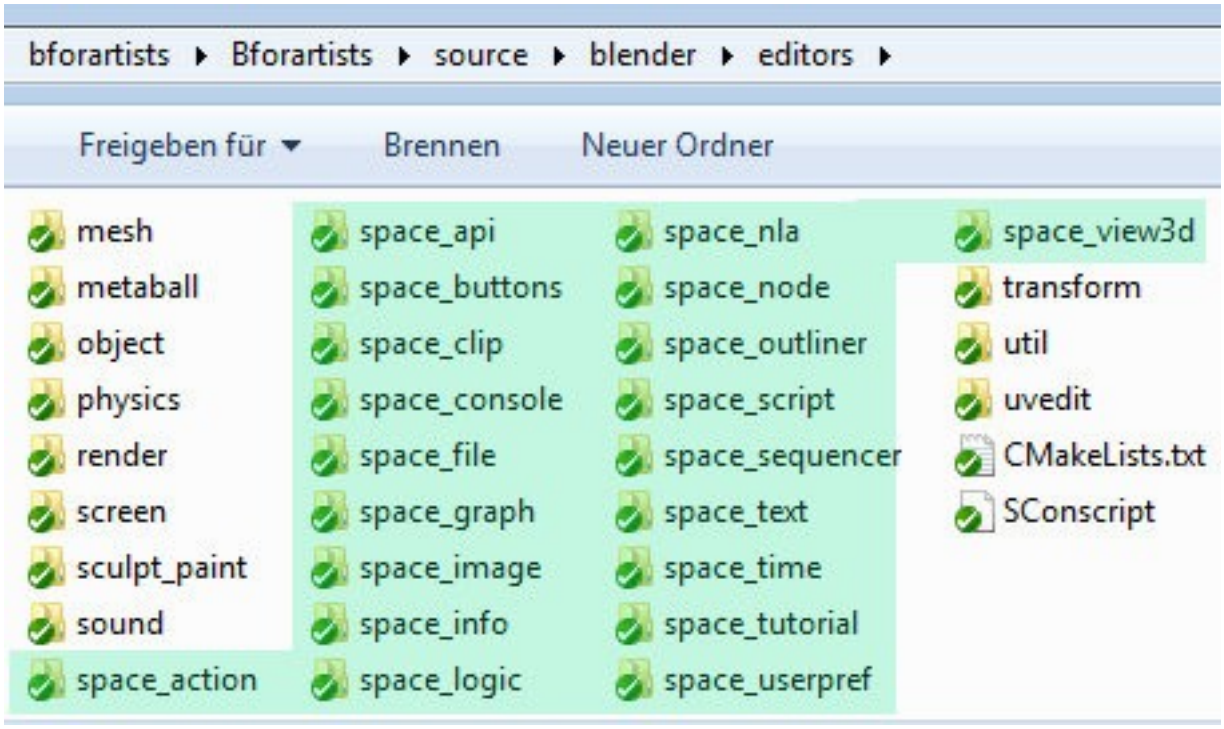
```
/* only call once on startup, storage is global in BKE kernel listbase */
```

```
void ED_spacetypes_init(void)
{
        const ListBase *spacetypes;
        SpaceType *type;

        /* UI_UNIT_X is now a variable, is used in some spacetype inits? */
        U.widget_unit = 20;

        /* create space types */
        ED_spacetype_outliner();
        ED_spacetype_time();
        ED_spacetype_view3d();
        ED_spacetype_ipo();
        ED_spacetype_image();
        ED_spacetype_node();
        ED_spacetype_buttons();
        ED_spacetype_info();
        ED_spacetype_file();
        ED_spacetype_action();
        ED_spacetype_nla();
        ED_spacetype_script();
        ED_spacetype_text();
        ED_spacetype_sequencer();
        ED_spacetype_logic();
        ED_spacetype_console();
        ED_spacetype_userpref();
        ED_spacetype_clip();
        ...
```

The declaration for each spacetype initialization function is put in blender/source/blender/editors/include/ED_space_api.h.

```
/* the pluginnable API for export to editors */

/* calls for registering default spaces */
void ED_spacetype_outliner(void);
void ED_spacetype_time(void);
void ED_spacetype_view3d(void);
void ED_spacetype_ipo(void);
void ED_spacetype_image(void);
void ED_spacetype_node(void);
void ED_spacetype_buttons(void);
void ED_spacetype_info(void);
void ED_spacetype_file(void);
void ED_spacetype_action(void);
void ED_spacetype_nla(void);
void ED_spacetype_script(void);
void ED_spacetype_text(void);
void ED_spacetype_tutorial(void);
void ED_spacetype_sequencer(void);
void ED_spacetype_logic(void);
void ED_spacetype_console(void);
void ED_spacetype_userpref(void);
void ED_spacetype_clip(void);
```

Each space type is implemented in a set of source files under blender/source/blender/editors. The following image highlights the directories associated with the various space types (editors).



Examining the logic space type contained in *space_logic/space_logic.c* reveals the definition of the *ED_spacetype_logic* function which is called by *ED_spacetypes_init()* as indicated above.

```
/* only called once, from space/spacetypes.c */
void ED_spacetype_logic(void)
{
	SpaceType *st = MEM_callocN(sizeof(SpaceType), "spacetype logic");
	ARegionType *art;

	st->spaceid = SPACE_LOGIC;
	strncpy(st->name, "Logic", BKE_ST_MAXNAME);

	st->new = logic_new;
	st->free = logic_free;
	st->init = logic_init;
	st->duplicate = logic_duplicate;
	st->operatortypes = logic_operatortypes;
	st->keymap = logic_keymap;
	st->refresh = logic_refresh;
	st->context = logic_context;

	/* regions: main window */
	art = MEM_callocN(sizeof(ARegionType), "spacetype logic region");
	art->regionid = RGN_TYPE_WINDOW;
	art->keymapflag = ED_KEYMAP_UI | ED_KEYMAP_FRAMES | ED_KEYMAP_VIEW2D;
	art->init = logic_main_area_init;
	art->draw = logic_main_area_draw;
	art->listener = logic_listener;

	BLI_addhead(&st->regiontypes, art);

	/* regions: listview/buttons */
	art = MEM_callocN(sizeof(ARegionType), "spacetype logic region");
	art->regionid = RGN_TYPE_UI;
	art->prefsizex= 220; // XXX
	art->keymapflag = ED_KEYMAP_UI | ED_KEYMAP_FRAMES;
	art->listener = logic_listener;
	art->init = logic_buttons_area_init;
	art->draw = logic_buttons_area_draw;
```

```
        BLI_addhead(&st->regiontypes, art);

        /* regions: header */
        art= MEM_callocN(sizeof(ARegionType), "spacetype logic region");
        art->regionid = RGN_TYPE_HEADER;
        art->prefsizey = HEADERY;
        art->keymapflag = ED_KEYMAP_UI | ED_KEYMAP_VIEW2D | ED_KEYMAP_FRAMES |
ED_KEYMAP_HEADER;
        art->init = logic_header_area_init;
        art->draw = logic_header_area_draw;

        BLI_addhead(&st->regiontypes, art);

        BKE_spacetype_register(st);
}
```

The top portion of this function allocates a new *SpaceType* structure and a new *RegionType* structure. The *st SpaceType* structure is updated with it's *spaceid* set to *SPACE_LOGIC*, a name and function pointers for each callback.

Region types are allocated for the main logic area, the header and the properties list. Each region type is added to the *regiontypes* linked list in the *st SpaceType* structure by calling *BLI_add_head()* which adds the region type to the head of the linked list.

Finally, the space type is registered with Blender with a call to *BKE_spacetype_register(st)* which essentially is a wrapper around *BLI_add_tail()* which first checks to make sure that the space type has not already been assigned.

The first time the user selects an editor from the Editor Type menu for a given space the *new* callback for the space type is called. The function returns a *SpaceLink* type which is a generic pointer for all of the specific space types.

Search for the *logic_new()* function in source/blender/editors/space_logic/space_logic.c.

```
static SpaceLink *logic_new(const bContext *C)
{
        ScrArea *sa= CTX_wm_area(C);
        ARegion *ar;
        SpaceLogic *slogic;

        slogic= MEM_callocN(sizeof(SpaceLogic), "initlogic");
        slogic->spacetype= SPACE_LOGIC;

        /* default options */
        slogic->scaflag = ((BUTS_SENS_SEL|BUTS_SENS_ACT|BUTS_SENS_LINK) |
                           (BUTS_CONT_SEL|BUTS_CONT_ACT|BUTS_CONT_LINK) |
                           (BUTS_ACT_SEL|BUTS_ACT_ACT|BUTS_ACT_LINK)    |
                           (BUTS_SENS_STATE|BUTS_ACT_STATE));


        /* header */
        ar= MEM_callocN(sizeof(ARegion), "header for logic");

        BLI_addtail(&slogic->regionbase, ar);
        ar->regiontype= RGN_TYPE_HEADER;
        ar->alignment= RGN_ALIGN_BOTTOM;

        /* buttons/list view */
        ar= MEM_callocN(sizeof(ARegion), "buttons for logic");
```

```
        BLI_addtail(&slogic->regionbase, ar);
        ar->regiontype= RGN_TYPE_UI;
        ar->alignment= RGN_ALIGN_LEFT;

        /* main area */
        ar= MEM_callocN(sizeof(ARegion), "main area for logic");

        BLI_addtail(&slogic->regionbase, ar);
        ar->regiontype= RGN_TYPE_WINDOW;

        ar->v2d.tot.xmin =  0.0f;
        ar->v2d.tot.ymax =  0.0f;
        ar->v2d.tot.xmax = 1150.0f;
        ar->v2d.tot.ymin = ( 1150.0f/(float)sa->winx ) * (float)-sa->winy;

        ar->v2d.cur = ar->v2d.tot;

        ar->v2d.min[0] = 1.0f;
        ar->v2d.min[1] = 1.0f;

        ar->v2d.max[0] = 32000.0f;
        ar->v2d.max[1] = 32000.0f;

        ar->v2d.minzoom = 0.5f;
        ar->v2d.maxzoom = 1.5f;

        ar->v2d.scroll = (V2D_SCROLL_RIGHT | V2D_SCROLL_BOTTOM);
        ar->v2d.keepzoom = V2D_KEEPZOOM | V2D_LIMITZOOM | V2D_KEEPASPECT;
        ar->v2d.keeptot = V2D_KEEPTOT_BOUNDS;
        ar->v2d.align = V2D_ALIGN_NO_POS_Y | V2D_ALIGN_NO_NEG_X;
        ar->v2d.keepofs = V2D_KEEPOFS_Y;

        return (SpaceLink *)slogic;
}
```

logic_new() allocates a new *SpaceLogic* structure and sets its *spaceid* to *SPACE_LOGIC*. Then it proceeds to create 3 regions for the main logic region, the header region and the properties region and adding these regions to the *regionbase* linked list within the space. Finally, the pointer to the *SpaceLogic* structure is returned as a *SpaceLink* pointer.

While there is much more going on in regard to initialization, keymaps etc in most of the files discussed so far we now have enough information to create a new space with minimal functionality.

Implementing the basic editor

Note

This part differs already from the above mentioned tutorial. There are quite a few changes. For example , editing editortype_pup() doesn't exist anymore. That's an RNA enum now.

The actual changes to implement a naked editor in Bforartists, which is based at Blender 2.76:

1 build_files/cmake/macros.cmake

@@ -512,6 +512,7 @@ function(SETUP_BLENDER_SORTED_LIBS)

```
                bf_editor_space_script
```

```
		bf_editor_space_sequencer
		bf_editor_space_text
+		bf_editor_space_tutorial
		bf_editor_space_time
		bf_editor_space_userpref
		bf_editor_space_view3d
```

1 source/blender/blenkernel/BKE_context.h

@@ -163,6 +162,7 @@ struct SpaceAction *CTX_wm_space_action(const bContext *C);

```
 struct SpaceInfo *CTX_wm_space_info(const bContext *C);
 struct SpaceUserPref *CTX_wm_space_userpref(const bContext *C);
 struct SpaceClip *CTX_wm_space_clip(const bContext *C);
+struct SpaceTutorial *CTX_wm_space_tutorial(const bContext *C); // bfa - our new tutorial editor

 void CTX_wm_manager_set(bContext *C, struct wmWindowManager *wm);
 void CTX_wm_window_set(bContext *C, struct wmWindow *win);
```

10 source/blender/blenkernel/intern/context.c

@@ -693,6 +693,16 @@ struct SpaceText *CTX_wm_space_text(const bContext *C)

```
 	return NULL;
 }

+// bfa - our new tutorial editor
+struct SpaceTutorial *CTX_wm_space_tutorial(const bContext *C)
+{
+	ScrArea *sa = CTX_wm_area(C);
+	if (sa && sa->spacetype == SPACE_TUTORIAL)
+		return sa->spacedata.first;
+	return NULL;
+}
+
+
 struct SpaceConsole *CTX_wm_space_console(const bContext *C)
 {
 	ScrArea *sa = CTX_wm_area(C);
```

1 source/blender/editors/CMakeLists.txt

@@ -50,6 +50,7 @@ if(WITH_BLENDER)

```
 	add_subdirectory(space_script)
 	add_subdirectory(space_sequencer)
```

```
        add_subdirectory(space_text)
+       add_subdirectory(space_tutorial)
        add_subdirectory(space_time)
        add_subdirectory(space_userpref)
        add_subdirectory(space_view3d)
```

1 source/blender/editors/include/ED_space_api.h

@@ -58,6 +58,7 @@ void ED_spacetype_logic(void);

```
 void ED_spacetype_console(void);
 void ED_spacetype_userpref(void);
 void ED_spacetype_clip(void);
+void ED_spacetype_tutorial(void);  // bfa - /* Our new space/editor */

 /* calls for instancing and freeing spacetype static data
  * called in WM_init_exit */
```

1 source/blender/editors/space_api/spacetypes.c

@@ -96,6 +96,7 @@ void ED_spacetypes_init(void)

```
        ED_spacetype_console();
        ED_spacetype_userpref();
        ED_spacetype_clip();
+       ED_spacetype_tutorial(); // bfa - our tutorial editor
 //     ...

        /* register operator types for screen and all spaces */
```

21 source/blender/editors/space_tutorial/CMakeLists.txt

This file is new created.

@@ -0,0 +1,21 @@

```
+set(INC
+       ../include
+       ../../blenkernel
+       ../../blenlib
+       ../../gpu
+       ../../makesdna
+       ../../makesrna
+       ../../windowmanager
+       ../../../../intern/guardedalloc
+       ../../../../intern/glew-mx
+)
+
+set(INC_SYS
+       ${GLEW_INCLUDE_PATH}
+)
+
+set(SRC
```

```
+        space_tutorial.c
+)
+
+blender_add_lib(bf_editor_space_tutorial "${SRC}" "${INC}" "${INC_SYS}")
```

151 source/blender/editors/space_tutorial/space_tutorial.c

This file is created new

@@ -0,0 +1,151 @@

```
+/*
+* ***** BEGIN GPL LICENSE BLOCK *****
+*
+* This program is free software; you can redistribute it and/or
+* modify it under the terms of the GNU General Public License
+* as published by the Free Software Foundation; either version 2
+* of the License, or (at your option) any later version.
+*
+* This program is distributed in the hope that it will be useful,
+* but WITHOUT ANY WARRANTY; without even the implied warranty of
+* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
+* GNU General Public License for more details.
+*
+* You should have received a copy of the GNU General Public License
+* along with this program; if not, write to the Free Software Foundation,
+* Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.
+*
+* The Original Code is Copyright (C) 2008 Blender Foundation.
+* All rights reserved.
+*
+*
+* Contributor(s): Michael Neilly
+*
+* ***** END GPL LICENSE BLOCK *****
+*/
+
+/** \file blender/editors/space_tutorial/space_tutorial.c
+*  \ingroup sptutorial
+*/
+
+#include <string.h>
+
+#include "DNA_text_types.h"
+
+#include "MEM_guardedalloc.h"
+
+#include "BLI_blenlib.h"
+
+#include "BKE_context.h"
+#include "BKE_screen.h"
+
```

```
+#include "ED_space_api.h"
+#include "ED_screen.h"
+
+#include "BIF_gl.h"
+
+#include "WM_api.h"
+#include "WM_types.h"
+
+#include "UI_interface.h"
+#include "UI_resources.h"
+#include "UI_view2d.h"
+
+static SpaceLink *tutorial_new(const bContext *C)
+{
+	ScrArea *sa = CTX_wm_area(C);
+	ARegion *ar;
+	SpaceTutorial *stutorial;
+
+	stutorial = MEM_callocN(sizeof(SpaceTutorial), "inittutorial");
+	stutorial->spacetype = SPACE_TUTORIAL;
+
+	/* header */
+	ar = MEM_callocN(sizeof(ARegion), "header for tutorial");
+
+	BLI_addtail(&stutorial->regionbase, ar);
+	ar->regiontype = RGN_TYPE_HEADER;
+	ar->alignment = RGN_ALIGN_BOTTOM;
+
+	/* main area */
+	ar = MEM_callocN(sizeof(ARegion), "main area for tutorial");
+
+	BLI_addtail(&stutorial->regionbase, ar);
+	ar->regiontype = RGN_TYPE_WINDOW;
+
+	return (SpaceLink *)stutorial;
+}
+
+/* add handlers, stuff you only do once or on area/region changes */
+static void tutorial_main_area_init(wmWindowManager *wm, ARegion *ar)
+{
+	UI_view2d_region_reinit(&ar->v2d, V2D_COMMONVIEW_CUSTOM, ar->winx, ar->winy);
+}
+
+static void tutorial_main_area_draw(const bContext *C, ARegion *ar)
+{
+	/* draw entirely, view changes should be handled here */
+	SpaceTutorial *stutorial = CTX_wm_space_tutorial(C);
+	View2D *v2d = &ar->v2d;
+	View2DScrollers *scrollers;
+
+	/* clear and setup matrix */
+	UI_ThemeClearColor(TH_BACK);
+	glClear(GL_COLOR_BUFFER_BIT);
+
+	/* works best with no view2d matrix set */
+	UI_view2d_view_ortho(v2d);
```

```
+
+	/* reset view matrix */
+	UI_view2d_view_restore(C);
+
+	/* scrollers */
+	scrollers = UI_view2d_scrollers_calc(C, v2d, V2D_ARG_DUMMY, V2D_ARG_DUMMY,
V2D_ARG_DUMMY, V2D_GRID_CLAMP);
+	UI_view2d_scrollers_draw(C, v2d, scrollers);
+	UI_view2d_scrollers_free(scrollers);
+}
+
+static void tutorial_header_area_init(wmWindowManager *UNUSED(wm), ARegion *ar)
+{
+	ED_region_header_init(ar);
+}
+
+static void tutorial_header_area_draw(const bContext *C, ARegion *ar)
+{
+	ED_region_header(C, ar);
+}
+
+/********************* registration ********************/
+
+/* only called once, from space/spacetypes.c */
+void ED_spacetype_tutorial(void)
+{
+	SpaceType *st = MEM_callocN(sizeof(SpaceType), "spacetype tutorial");
+	ARegionType *art;
+
+	st->spaceid = SPACE_TUTORIAL;
+	strncpy(st->name, "Tutorial", BKE_ST_MAXNAME);
+
+	st->new = tutorial_new;
+
+	/* regions: main window */
+	art = MEM_callocN(sizeof(ARegionType), "spacetype tutorial region");
+	art->regionid = RGN_TYPE_WINDOW;
+
+	art->init = tutorial_main_area_init;
+	art->draw = tutorial_main_area_draw;
+
+	BLI_addhead(&st->regiontypes, art);
+
+	/* regions: header */
+	art = MEM_callocN(sizeof(ARegionType), "spacetype tutorial region");
+	art->regionid = RGN_TYPE_HEADER;
+	art->prefsizey = HEADERY;
+	art->keymapflag = ED_KEYMAP_UI | ED_KEYMAP_VIEW2D | ED_KEYMAP_HEADER;
+	art->init = tutorial_header_area_init;
+	art->draw = tutorial_header_area_draw;
+
+	BLI_addhead(&st->regiontypes, art);
+
+	BKE_spacetype_register(st);
+}
```

13 source/blender/makesdna/DNA_space_types.h

@@ -1324,6 +1324,15 @@ typedef enum eSpaceClip_GPencil_Source {

```
        SC_GPENCIL_SRC_TRACK = 1,
 } eSpaceClip_GPencil_Source;

+
+/* Tutorial Editor */  // bfa - the space link for our tutorial editor
+typedef struct SpaceTutorial {
+       SpaceLink *next, *prev;
+       ListBase regionbase;
+       int spacetype;
+       char pad[4];
+} SpaceTutorial;
+
 /* ***************** SPACE DEFINES ********************** */

 /* space types, moved from DNA_screen_types.h */
@@ -1353,8 +1362,10 @@ typedef enum eSpace_Type {
        SPACE_CONSOLE  = 18,
        SPACE_USERPREF = 19,
        SPACE_CLIP     = 20,
+       SPACE_TUTORIAL = 21, // bfa our tutorial space type

-       SPACEICONMAX = SPACE_CLIP
+       SPACEICONMAX = SPACE_TUTORIAL // bfa - space clip repaced by space tutorial
+       //SPACEICONMAX
 } eSpace_Type;

 /* use for function args */
```

20 source/blender/makesrna/intern/rna_space.c

Pay attention to the double entry at the end of the enum. **{ 0, NULL, 0, NULL, NULL }, .** Withbout this entry the last menu item doesn't show.

@@ -85,8 +85,10 @@ EnumPropertyItem space_type_items[] = {

```
        {0, "", ICON_NONE, NULL, NULL},
        {SPACE_CONSOLE, "CONSOLE", ICON_CONSOLE, "Python Console", "Interactive programmatic console for advanced editing and script development"},
        { 0, "", ICON_NONE, NULL, NULL },
        {SPACE_SEQ, "SEQUENCE_EDITOR", ICON_SEQUENCE, "DEPRECATED - VSE", "Video editing tools. DEPRECATED. USE AT OWN RISK."}, // Deprecated video sequence editor
+       { SPACE_TUTORIAL, "TUTORIAL_EDITOR", ICON_SEQUENCE, "Tutorial Editor", "Tooltip" },
        { 0, NULL, 0, NULL, NULL },
+       { 0, NULL, 0, NULL, NULL },
 };

 #define V3D_S3D_CAMERA_LEFT       {STEREO_LEFT_ID, "LEFT",
```

```
ICON_RESTRICT_RENDER_OFF, "Left", ""},
```

@@ -297,6 +299,8 @@ static StructRNA *rna_Space_refine(struct PointerRNA *ptr)

```
                        return &RNA_SpaceSequenceEditor;
                case SPACE_TEXT:
                        return &RNA_SpaceTextEditor;
+               case SPACE_TUTORIAL: // bfa - the new tutorial editor
+                       return &RNA_SpaceTutorialEditor;
                case SPACE_ACTION:
                        return &RNA_SpaceDopeSheetEditor;
                case SPACE_NLA:
```

@@ -3334,6 +3338,19 @@ static void rna_def_space_text(BlenderRNA *brna)

Pay attention to the **//PropertyRNA *prop; line.** I had to comment it out. VS threw an error here. Referenced before assigned. So i have turned it off for now.

```
        RNA_api_space_text(srna);
 }

+// bfa - our new tutorial editor
+static void rna_def_space_tutorial(BlenderRNA *brna)
+{
+       StructRNA *srna;
+       //PropertyRNA *prop;
+
+       srna = RNA_def_struct(brna, "SpaceTutorialEditor", "Space");
+       RNA_def_struct_sdna(srna, "SpaceTutorial");
+       RNA_def_struct_ui_text(srna, "Space Tutorial Editor", "Tutorial editor space data");
+
+}
+
+
 static void rna_def_space_dopesheet(BlenderRNA *brna)
 {
        StructRNA *srna;
```

@@ -4737,6 +4754,7 @@ void RNA_def_space(BlenderRNA *brna)

```
        rna_def_space_node(brna);
        rna_def_space_logic(brna);
        rna_def_space_clip(brna);
+       rna_def_space_tutorial(brna); // bfa - our new tutorial editor
 }

 #endif
```

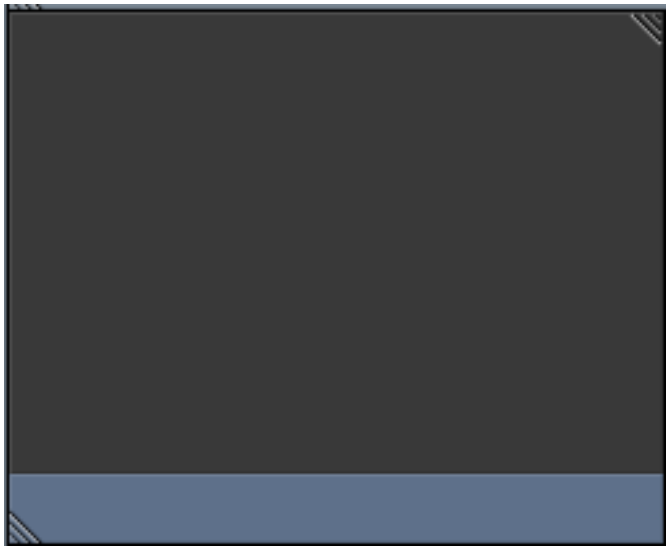
source/blender/windowmanager/WM_types.h

@@ -362,6 +362,7 @@ typedef struct wmNotifier {

```
 #define ND_SPACE_CHANGED          (18<<16) /*sent to a new editor type after it's replaced an old one*/
 #define ND_SPACE_CLIP             (19<<16)
 #define ND_SPACE_FILE_PREVIEW     (20<<16)
+#define ND_SPACE_TUTORIAL          (21<<16) // bfa - the new tutorial editor

 /* subtype, 256 entries too */
 #define NOTE_SUBTYPE              0x0000FF00
```

The Python files for the menu

Here we go, we should have this after the above steps.



The current editor is naked. It is fully functional. You can resize it, drag-open another editor, etc. But it has no menu and no other content. That's what we want to change now.

In this chapter we want to add a menu to the header. That's done with two python files.

release/scripts/startup/bl_ui/space_tutorial.py

Let's start at the end. The python files that defines the content of the menus.

The editors all have a python file in the release\scripts\startup\bl_ui\ path . space_info.py for example.

That's where things like menus and panel content gets defined. We could place our needed content in one of the existing files. But it's nicer to do this in its own file. Let's create a space_tutorial.py file.

I have grabbed the space_info.py file here, and removed everything except one menu and one menu item. The classes are also renamed already to fit to our tutorial space. This file is reduced to the bones.

Python is intendation sensitive. Be careful here.

```python
import bpy
from bpy.types import Header, Menu


class TUTORIAL_HT_header(Header):
    bl_space_type = 'TUTORIAL'

    def draw(self, context):
        layout = self.layout

        window = context.window
        scene = context.scene
```

```
        ALL_MT_editormenu.draw_hidden(context, layout) # bfa - show hide the editormenu
        TUTORIAL_MT_editor_menus.draw_collapsible(context, layout)

# bfa - show hide the editormenu
class ALL_MT_editormenu(Menu):
    bl_label = ""

    def draw(self, context):
        self.draw_menus(self.layout, context)

    @staticmethod
    def draw_menus(layout, context):

        row = layout.row(align=True)
        row.template_header() # editor type menus

# ------------------------------menu items
# Everything menu in this class is collapsible. See line 35.
class TUTORIAL_MT_editor_menus(Menu):
    bl_idname = "TUTORIAL_MT_editor_menus"
    bl_label = ""

    def draw(self, context):
        self.draw_menus(self.layout, context)

    @staticmethod
    def draw_menus(layout, context):
        scene = context.scene
        rd = scene.render

        layout.menu("TUTORIAL_MT_file") # see class TUTORIAL_MT_file below

class TUTORIAL_MT_file(Menu):
    bl_label = "File"

    def draw(self, context):
        layout = self.layout

        layout.operator_context = 'INVOKE_AREA'
        layout.operator("wm.read_homefile", text="New", icon='NEW')

if __name__ == "__main__":  # only for live edit.
    bpy.utils.register_module(__name__)
```

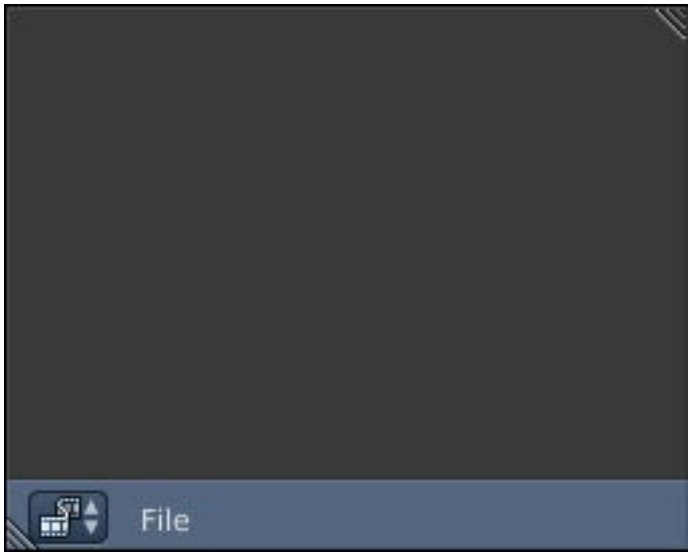
The problem is, it doesn't show yet when you place this file in the release\scripts\startup\bl_ui\ path. There is one more entry missing. The one in the __init__.py so that Blender knows our tutorial space exists.

release/scripts/startup/bl_ui/__init__.py

@@ -78,6 +78,8 @@

```
   "space_userpref",
   "space_view3d",
   "space_view3d_toolbar",
+  "space_tutorial",
 ]

 import bpy
```

And when everything worked as thought, then we should have this now:



Adding the icon

We have used the vse icon above. But our new tutorial editor needs its own icon. For that you need to modify the blender_icons.svg in the release/datafiles folder. It's a vector graphics sprite sheet that contains all icons for Blender. You need to have Inkscape installed. And a version of Blender (2.76 at this point) installed. Zipfile version is not enough. You might also add the path to the exe to the system variables. And you need some modifications at some other files. Then you run the blender_icons_update.py file, and the icons gets updated.

But one step after another.

First create a icon. And place it into the blender_icons.svg. As told, you need Inkscape here. And it must be a vector graphics icon. It's not done with throwing in a png file here.

blender_icons.svg

The blender_icons.svg is in \release\datafiles

Now let's do the changes at the files.

2 source/blender/editors/include/UI_icons.h

Here the icons gets defined. We remove the blank icon slot, and add our icon here.

@@ -171,9 +171,9 @@ DEF_ICON(CONSOLE)

```
 DEF_ICON(PREFERENCES)
 DEF_ICON(CLIP)
 DEF_ICON(ASSET_MANAGER)
+DEF_ICON(TUTORIAL)

 #ifndef DEF_ICON_BLANK_SKIP
-        DEF_ICON(BLANK057)
         DEF_ICON(BLANK058)
         DEF_ICON(BLANK059)
         DEF_ICON(BLANK060)
         DEF_ICON(BLANK061)
```

4 source/blender/makesrna/intern/rna_space.c

@@ -85,8 +85,8 @@ EnumPropertyItem space_type_items[] = {

```
         {0, "", ICON_NONE, NULL, NULL},
         {SPACE_CONSOLE, "CONSOLE", ICON_CONSOLE, "Python Console", "Interactive programmatic console for advanced editing and script development"},
         { 0, "", ICON_NONE, NULL, NULL },
-        { SPACE_TUTORIAL, "TUTORIAL_EDITOR", ICON_SEQUENCE, "Tutorial Editor", "Tooltip" },
+        { SPACE_TUTORIAL, "TUTORIAL_EDITOR", ICON_TUTORIAL, "Tutorial Editor", "Tooltip" },
         { 0, "", ICON_NONE, NULL, NULL },
```

4 source/blender/makesrna/intern/rna_userdef.c

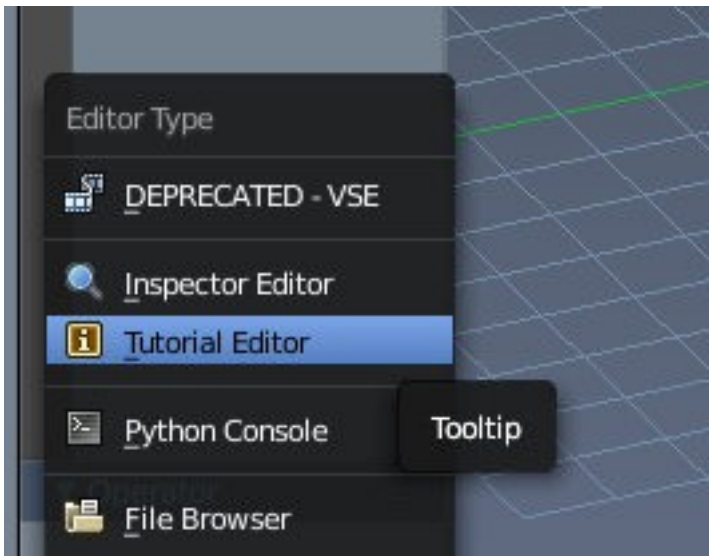
@@ -3033,8 +3033,8 @@ static void rna_def_userdef_themes(BlenderRNA *brna)

```
                 {16, "FILE_BROWSER", ICON_FILESEL, "File Browser", ""},
                 {17, "CONSOLE", ICON_CONSOLE, "Python Console", ""},
                 {20, "CLIP_EDITOR", ICON_CLIP, "Movie Clip Editor", ""},
-                {21, "TUTORIAL_EDITOR", ICON_CLIP, "Tutorial Editor", "" }, // bfa - our new tutorial editor
+                { 21, "TUTORIAL_EDITOR", ICON_TUTORIAL, "Tutorial Editor", "" }, // bfa - our new tutorial editor
                 {0, NULL, 0, NULL, NULL}
         };
```

Now let's create and update the icons. Open command line. Switch to the directory with the source code. I have it at drive h: . The tutorial is made with Bforartists. So i run this commands here. When you work with Blender then you have to change the path to your actual folder names:

```
h:

cd bforartists\bforartists\release\datafiles

python blender_icons_update.py blender_icons.svg
```

The last step is to compile, and the Tutorial editor should have its own icon now:



What you can also see, in case you want to modify the tooltip, 4 source/blender/makesrna/intern/rna_space.c is the file to do so ...

More changes

Our editor shows, has its icon, and has a first menu. But there are a few more loose ends around. Content and theming is missing. So let's have a look how the other editors are implemented.

A search for the terms "SpaceTime" and "SPACE_TIME" to have something to compare brought up a few more places where our new editor needs to have entries.

3 source/blender/blenloader/intern/writefile.c

@@ -2867,6 +2867,9 @@ static void write_screens(WriteData *wd, ListBase *scrbase)

```
                                else if (sl->spacetype == SPACE_INFO) {
                                        writestruct(wd, DATA, "SpaceInfo", 1, sl);
                                }
+                               else if (sl->spacetype == SPACE_TUTORIAL) {
+                                       writestruct(wd, DATA, "SpaceTutorial", 1, sl); /*bfa - second part tutorial editor*/
+                               }

                                sl= sl->next;
                        }
```

1 source/blender/makesrna/RNA_access.h

@@ -576,6 +576,7 @@ extern StructRNA RNA_SpaceUVEditor;

```
 extern StructRNA RNA_SpaceUserPreferences;
 extern StructRNA RNA_SpaceView3D;
 extern StructRNA RNA_SpaceClipEditor;
+extern StructRNA RNA_SpaceTutorialEditor; /*bfa - second part tutorial editor*/
 extern StructRNA RNA_Speaker;
 extern StructRNA RNA_SpeedControlSequence;
```

```
extern StructRNA RNA_Spline;
```

1 source/blender/python/intern/bpy_rna_callback.c

@@ -174,6 +174,7 @@ static eSpace_Type rna_Space_refine_reverse(StructRNA *srna)

```
        if (srna == &RNA_SpaceConsole)          return SPACE_CONSOLE;
        if (srna == &RNA_SpaceUserPreferences)  return SPACE_USERPREF;
        if (srna == &RNA_SpaceClipEditor)       return SPACE_CLIP;
+       if (srna == &RNA_SpaceTutorialEditor)   return SPACE_TUTORIAL; // bfa - second part tutorial editor
        return -1;
 }
```

3 source/blender/python/simple_enum_gen.py

@@ -36,6 +36,7 @@

```
    SPACE_SCRIPT, #Deprecated
    SPACE_TIME,
    SPACE_NODE,
+   SPACE_TUTORIAL # bfa - second part tutorial editor
    SPACEICONMAX
 """
```

Theming entries first steps

Some first steps:

6 source/blender/editors/interface/resources.c

@@ -165,6 +165,12 @@ const unsigned char *UI_ThemeGetColorPtr(bTheme *btheme, int spacetype, int colo

```
                                case SPACE_CLIP:
                                        ts = &btheme->tclip;
                                        break;
+                               case SPACE_TUTORIAL: // bfa - our new tutorial editor
+                                       ts = &btheme->ttutorial;
+                                       break;
                                default:
                                        ts = &btheme->tv3d;
                                        break;
```

5 source/blender/makesdna/DNA_userdef_types.h

@@ -386,6 +386,9 @@ typedef struct bTheme {

```
        ThemeSpace tuserpref;
        ThemeSpace tconsole;
        ThemeSpace tclip;
+       /*bfa - our new editor type tutorial*/
+       ThemeSpace ttutorial;
        /* 20 sets of bone colors for this theme */
        ThemeWireColor tarm[20];
```

@@ -395,7 +398,7 @@ typedef struct bTheme {

```
 } bTheme;

#define UI_THEMESPACE_START(btheme)  (CHECK_TYPE_INLINE(btheme, bTheme *),  &((btheme)->tbuts))
-#define UI_THEMESPACE_END(btheme)    (CHECK_TYPE_INLINE(btheme, bTheme *), (&((btheme)->tclip) + 1))
+#define UI_THEMESPACE_END(btheme)    (CHECK_TYPE_INLINE(btheme, bTheme *), (&((btheme)->ttutorial) + 1))

 /* for the moment only the name. may want to store options with this later */
 typedef struct bAddon {
```

6 source/blender/makesrna/RNA_access.h

@@ -576,8 +576,8 @@ extern StructRNA RNA_SpaceUVEditor;

```
 extern StructRNA RNA_SpaceUserPreferences;

 extern StructRNA RNA_SpaceView3D;

 extern StructRNA RNA_SpaceClipEditor;

-extern StructRNA RNA_SpaceTutorialEditor; /*bfa - second part tutorial editor*/

+extern StructRNA RNA_SpaceTutorialEditor;

 extern StructRNA RNA_Speaker;

 extern StructRNA RNA_SpeedControlSequence;

 extern StructRNA RNA_Spline;
```

@@ -650,6 +650,8 @@ extern StructRNA RNA_ThemeTimeline;

```
 extern StructRNA RNA_ThemeUserInterface;

 extern StructRNA RNA_ThemeUserPreferences;

 extern StructRNA RNA_ThemeView3D;

+extern StructRNA RNA_ThemeTutorialEditor;
```

```
extern StructRNA RNA_ThemeWidgetColors;
extern StructRNA RNA_ThemeWidgetStateColors;
extern StructRNA RNA_TimelineMarker;
```

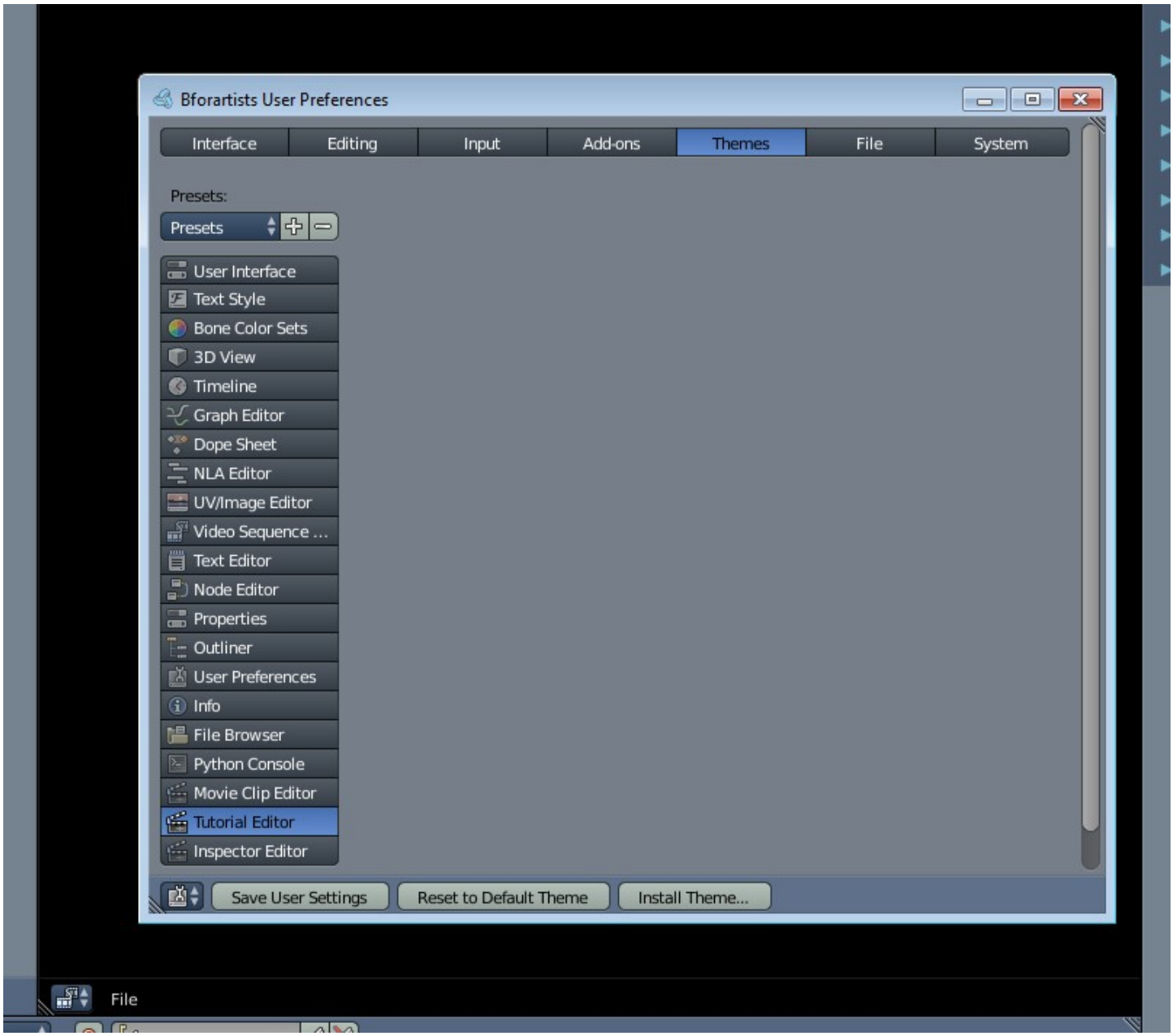
16 source/blender/makesrna/intern/rna_userdef.c

@@ -3033,6 +3033,8 @@ static void rna_def_userdef_themes(BlenderRNA *brna)

```
                {16, "FILE_BROWSER", ICON_FILESEL, "File Browser", ""},
                {17, "CONSOLE", ICON_CONSOLE, "Python Console", ""},
                {20, "CLIP_EDITOR", ICON_CLIP, "Movie Clip Editor", ""},
+               {21, "TUTORIAL_EDITOR", ICON_CLIP, "Tutorial Editor", "" }, // bfa - our new tutorial editor
                {0, NULL, 0, NULL, NULL}
        };
```

And after this changes our new editor turns black. So we might be at the right track here.

What is missing is the theming part in the User Preferences. So that you can actually change the colors of our new editor.



Theming entries Second part

Note!

The following part of this document may or may not be valid. USE AT OWN RISK!

It is based at the work of somebody who gave me a loose collection of code snippets and files with vague descriptions instead of the whole project folder. And every time i asked i got a few more missing files. And in the end i had to give up at the content part because of too much not by me solvable errors.

Well, i got at least the theming working, which is this chapter about here. But with some odd workarounds to eliminate a persistent error. See (Bug)Fix for wrong RNA name. And i am not sure if i haven't broken already something for the content part.

As told above, what is missing is the rest of the theming. Note that you may also find content related steps already. This also includes keymap items.

Let's go through the necessary steps.

release/scripts/modules/bpy_extras/keyconfig_utils.py

@@ -132,6 +132,7 @@

```
    ('Standard Modal Map', 'EMPTY', 'WINDOW', []),
    ('Transform Modal Map', 'EMPTY', 'WINDOW', []),
+    ('Tutorial', 'TUTORIAL_EDITOR', 'WINDOW', []), #  -tutorial editor
    ]
```

release/scripts/startup/bl_ui/space_tutorial.py

@@ -18,7 +18,8 @@

```
 # <pep8 compliant>
 import bpy
-from bpy.types import Header, Menu
+from bpy.types import Header, Panel, Menu, Window
+from rna_prop_ui import PropertyPanel


 class TUTORIAL_HT_header(Header):
```

source/blender/blenkernel/intern/context.c

@@ -693,24 +693,6 @@

```
        return NULL;
 }

-// bfa - our new tutorial editor
-struct SpaceTutorial *CTX_wm_space_tutorial(const bContext *C)
-{
-       ScrArea *sa = CTX_wm_area(C);
-       if (sa && sa->spacetype == SPACE_TUTORIAL)
-               return sa->spacedata.first;
-       return NULL;
-}

 struct SpaceConsole *CTX_wm_space_console(const bContext *C)
 {
```

@@ -828,6 +810,23 @@

```
 {
        ScrArea *sa = CTX_wm_area(C);
        if (sa && sa->spacetype == SPACE_CLIP)
+               return sa->spacedata.first;
+       return NULL;
+}
+
+/*bfa - Tutorial editor */
+struct SpaceTutorial *CTX_wm_space_tutorial(const bContext *C)
+{
+       ScrArea *sa = CTX_wm_area(C);
+       if (sa && sa->spacetype == SPACE_TUTORIAL)
+               return sa->spacedata.first;
+       return NULL;
+}
```

source/blender/blenloader/intern/readfile.c

@@ -6245,6 +6245,26 @@

```
                                                }
                                        }
                                }
+
+                               /*bfa - tutorial editor */
+                               else if (sl->spacetype == SPACE_TUTORIAL) {
+                                       SpaceTutorial *stutorial = (SpaceTutorial *)sl;
+                                       stutorial->pinid = newlibadr(fd, sc->id.lib, stutorial->pinid);
+                                       if (stutorial->pinid == NULL) {
+                                               stutorial->flag &= ~TUTORIAL_PIN_CONTEXT;
+                                       }
+                               }
+
+
                                else if (sl->spacetype == SPACE_OUTLINER) {
                                        SpaceOops *so= (SpaceOops *)sl;
                                        so->search_tse.id = newlibadr(fd, NULL, so->search_tse.id);
```

@@ -6599,6 +6619,35 @@

```
                                        SCRIPT_SET_NULL(scpt->script);
                                }
                        }
+
+                       /*bfa - tutorial editor*/
+                       else if (sl->spacetype == SPACE_TUTORIAL) {
+                               SpaceTutorial *stutorial = (SpaceTutorial *)sl;
+                               stutorial->pinid = restore_pointer_by_name(newmain, stutorial->pinid, USER_IGNORE);
+                               //sobj->pinid = newlibadr(fd, sc->id.lib, sobj->pinid);
+                               if (stutorial->pinid == NULL) {
+                                       stutorial->flag &= ~TUTORIAL_PIN_CONTEXT;
+                               }
```

```
+                                       /* TODO: restore path pointers: T40046
+                                       * (complicated because this contains data pointers too, not just ID)*/
+                                       MEM_SAFE_FREE(stutorial->path);
+                               }
+                               ////////////////
+
                                else if (sl->spacetype == SPACE_OUTLINER) {
                                        SpaceOops *so= (SpaceOops *)sl;
```

@@ -6910,6 +6959,29 @@

```
                                        snla->ads = newdataadr(fd, snla->ads);
                                }
+
+                               /*bfa - tutorial editor */
+                               else if (sl->spacetype == SPACE_TUTORIAL) {
+                                       SpaceTutorial *stutorial = (SpaceTutorial *)sl;
+
+                                       stutorial->path = NULL;
+                                       //sbuts->texuser = NULL;
+                                       stutorial->mainbo = stutorial->mainb;
+                                       stutorial->mainbuser = stutorial->mainb;
+                               }
+                               ///////////////
+
                                else if (sl->spacetype == SPACE_OUTLINER) {
                                        SpaceOops *soops = (SpaceOops *) sl;
```

source/blender/editors/include/ED_screen.h

@@ -158,6 +158,10 @@

```
 int    ED_operator_logic_active(struct bContext *C);
 int    ED_operator_info_active(struct bContext *C);
 int    ED_operator_console_active(struct bContext *C);
+int    ED_operator_tutorial_active(struct bContext *C); /*bfa - tutorial editor */
+int    ED_operator_tutorial_active_no_editobject(struct bContext *C);/*bfa - tutorial editor */

 int    ED_operator_object_active(struct bContext *C);
```

source/blender/editors/interface/interface_panel.c

@@ -130,6 +130,17 @@

```
                        return BUT_VERTICAL;
                else if (ELEM(ar->regiontype, RGN_TYPE_UI, RGN_TYPE_TOOLS,
RGN_TYPE_TOOL_PROPS))
                        return BUT_VERTICAL;
+
+               /*bfa - tutorial editor */
+               else if (sa->spacetype == SPACE_TUTORIAL && ar->regiontype == RGN_TYPE_WINDOW) {
+                       SpaceTutorial *stutorial = sa->spacedata.first;
+                       return stutorial->align;
```

```
+       }

        return 0;
 }
```

@@ -152,6 +163,24 @@

```
                return 1;
        else if (sa->spacetype == SPACE_FILE && ar->regiontype == RGN_TYPE_CHANNELS)
                return 1;
+       /*bfa - tutorial editor */
+       else if (sa->spacetype == SPACE_TUTORIAL && ar->regiontype == RGN_TYPE_WINDOW) {
+               SpaceTutorial *stutorial = sa->spacedata.first;
+
+               if (stutorial->align)
+                       if (stutorial->re_align || stutorial->mainbo != stutorial->mainb)
+                               return 1;
+       }
+
+

        /* in case panel is added or disappears */
        for (pa = ar->panels.first; pa; pa = pa->next) {
```

source/blender/editors/interface/resources.c

@@ -1103,6 +1103,22 @@

```
        rgba_char_args_set(btheme->text.shade1,     143, 143, 143, 255);
        rgba_char_args_set(btheme->text.shade2,     0xc6, 0x77, 0x77, 255);
        rgba_char_args_set(btheme->text.hilite,     255, 0, 0, 255);
+
+       /* bfa - space tutorial */
+       btheme->ttutorial = btheme->tv3d;
+       rgba_char_args_set_fl(btheme->ttutorial.back, 0.45, 0.45, 0.45, 1.0);
+
+       rgba_char_args_set_fl(btheme->ttutorial.match, 0.2, 0.5, 0.2, 0.3);    /* highlighting search match
- soft green*/
+       rgba_char_args_set_fl(btheme->ttutorial.selected_highlight, 0.51, 0.53, 0.55, 0.3);
+

        /* syntax highlighting */
        rgba_char_args_set(btheme->text.syntaxn,    0, 0, 200, 255);    /* Numbers  Blue*/
```

source/blender/editors/screen/screen_ops.c

@@ -326,6 +326,47 @@

```
        return ed_spacetype_test(C, SPACE_CONSOLE);
 }

+/*-----------------------------------------------------------------------------*/
+/*bfa - tutorial editor */
```

```
+int ED_operator_tutorial_active(bContext *C)
+{
+	return ed_spacetype_test(C, SPACE_TUTORIAL);
+}
+
+int ED_operator_tutorial_active_no_editobject(bContext *C)
+{
+	if (ed_spacetype_test(C, SPACE_TUTORIAL)) {
+		Object *ob = ED_object_active_context(C);
+		Object *obedit = CTX_data_edit_object(C);
+		if (ob && ob == obedit)
+			return 0;
+		else
+			return 1;
+	}
+	return 0;
+}
+
+
+/*---------------------------------------------------------------------------*/
+
 static int ed_object_hidden(Object *ob)
 {
	/* if hidden but in edit mode, we still display, can happen with animation */
```

source/blender/makesdna/DNA_space_types.h

@@ -306,6 +306,97 @@

```
	SO_SEARCH_RECURSIVE	= (1 << 2),
 } eSpaceOutliner_Search_Flags;

+///* Tutorial Editor */  // bfa - the space link for our tutorial editor
+//typedef struct SpaceTutorial {
+//	SpaceLink *next, *prev;
+//	ListBase regionbase;
+//	int spacetype;
+//	char pad[4];
+//} SpaceTutorial;
+//
+
+/* Tutorial Editor ======================================== */
+
+typedef struct SpaceTutorial {
+	SpaceLink *next, *prev;
+	ListBase regionbase;		/* storage of regions for inactive spaces */
+	int spacetype;
+	float blockscale DNA_DEPRECATED;
+	short blockhandler[8]  DNA_DEPRECATED;
+
+	View2D v2d DNA_DEPRECATED;  /* deprecated, copied to region */
+
+	short mainb, mainbo, mainbuser; /* context tabs */
+	short re_align, align;		/* align for panels */
```

```
+	short preview;			/* preview is signal to refresh */
+	/* texture context selector (material, lamp, particles, world, other)*/
+	short texture_context, texture_context_prev;
+	char flag, pad[7];
+
+	void *path;			/* runtime */
+	int pathflag, dataicon;		/* runtime */
+	ID *pinid;
+
+	void *texuser;
+} SpaceTutorial;
+
+/* SpaceTutorial->flag */
+typedef enum eSpaceTutorial_Flag {
+	TUTORIAL_PIN_CONTEXT = (1 << 1),
+} eSpaceTutorial_Flag;
+
+/* sbuts->align */
+typedef enum eSpaceTutorial_Align {
+	TUTORIAL_FREE = 0,
+	TUTORIAL_HORIZONTAL = 1,
+	TUTORIAL_VERTICAL = 2,
+	TUTORIAL_AUTO = 3,
+} eSpaceTutorial_Align;
+
 /* Graph Editor ========================================= */
```

@@ -1325,21 +1416,7 @@

```
 } eSpaceClip_GPencil_Source;


-/* Tutorial Editor */  // bfa - the space link for our tutorial editor
-typedef struct SpaceTutorial {
-	SpaceLink *next, *prev;
-	ListBase regionbase;
-	int spacetype;
-	char pad[4];
-} SpaceTutorial;


 /* ***************** SPACE DEFINES ********************** */
```

source/blender/makesrna/intern/rna_space.c

@@ -85,8 +85,8 @@
Just some cosmetics. Tooltip ...

```
	{0, "", ICON_NONE, NULL, NULL},
	{SPACE_CONSOLE, "CONSOLE", ICON_CONSOLE, "Python Console", "Interactive programmatic console for advanced editing and script development"},
	{ 0, "", ICON_NONE, NULL, NULL },
-	{ SPACE_TUTORIAL, "TUTORIAL_EDITOR", ICON_OBJECTINFO, "Tutorial Editor", "Tooltip" },
```

```
+	{ SPACE_TUTORIAL, "TUTORIAL_EDITOR", ICON_OBJECTINFO, "Tutorial Editor",
"Tooltip for Tutorialeditor" },
	{ 0, "", ICON_NONE, NULL, NULL },
	{ SPACE_SEQ, "SEQUENCE_EDITOR", ICON_SEQUENCE, "DEPRECATED - VSE", "Video
editing tools. DEPRECATED. USE AT OWN RISK." }, // Deprecated video sequence editor
	{ 0, NULL, 0, NULL, NULL },
```

@@ -318,9 +318,9 @@
Bugfix for wrong RNA name. RNA_SpaceTutorialEditor is wrong. RNA_ThemeTutorialEditor is right

```
		case SPACE_CLIP:
			return &RNA_SpaceClipEditor;
		case SPACE_TUTORIAL: // bfa - the new tutorial editor
-			return &RNA_SpaceTutorialEditor;
+			return &RNA_ThemeTutorialEditor;
		case SPACE_INSPECTOR: // bfa - the new inspector editor

		default:
			return &RNA_Space;
	}
```

@@ -984,6 +984,254 @@

```
		WM_main_add_notifier(NC_TEXT | NA_EDITED, st->text);
 }

+/*-----------------------------------------------------------------------------------------------------*/
+
+/* bfa  - Space tutorial */
+
+static void rna_SpaceTutorial_align_set(PointerRNA *ptr, int value)
+{
+	SpaceTutorial *stutorial = (SpaceTutorial *)(ptr->data);
+
+	stutorial->align = value;
+	stutorial->re_align = 1;
+}
+
+/* note: this function exists only to avoid id refcounting */
+static void rna_SpaceTutorial_pin_id_set(PointerRNA *ptr, PointerRNA value)
+{
+	SpaceTutorial *stutorial = (SpaceTutorial *)(ptr->data);
+	stutorial->pinid = value.data;
+}
+
+static StructRNA *rna_SpaceTutorial_pin_id_typef(PointerRNA *ptr)
+{
+	SpaceTutorial *stutorial = (SpaceTutorial *)(ptr->data);
+
+	if (stutorial->pinid)
+		return ID_code_to_RNA_type(GS(stutorial->pinid->name));
+
+	return &RNA_ID;
+}
```

```
+
+static void rna_SpaceTutorial_pin_id_update(Main *UNUSED(bmain), Scene *UNUSED(scene),
PointerRNA *ptr)
+{
+	SpaceTutorial *stutorial = (SpaceTutorial *)(ptr->data);
+	ID *id = stutorial->pinid;
+
+	if (id == NULL) {
+		stutorial->flag &= ~TUTORIAL_PIN_CONTEXT;
+		return;
+	}
+
+	switch (GS(id->name)) {
+	case ID_MA:
+		WM_main_add_notifier(NC_MATERIAL | ND_SHADING, NULL);
+		break;
+	case ID_TE:
+		WM_main_add_notifier(NC_TEXTURE, NULL);
+		break;
+	case ID_WO:
+		WM_main_add_notifier(NC_WORLD, NULL);
+		break;
+	case ID_LA:
+		WM_main_add_notifier(NC_LAMP, NULL);
+		break;
+	}
+}
+
+
+static void rna_SpaceTutorial_context_set(PointerRNA *ptr, int value)
+{
+	SpaceTutorial *stutorial = (SpaceTutorial *)(ptr->data);
+
+	stutorial->mainb = value;
+	stutorial->mainbuser = value;
+}
+
+static EnumPropertyItem *rna_SpaceTutorial_context_itemf(bContext *UNUSED(C), PointerRNA
*ptr,
+	PropertyRNA *UNUSED(prop), bool *r_free)
+{
+	SpaceTutorial *stutorial = (SpaceTutorial *)(ptr->data);
+	EnumPropertyItem *item = NULL;
+	int totitem = 0;
+	/*bfa - later step*/
+	//stutorial->pathflag & (1 << OBJCONTEXT_TUTORIAL);
+	//RNA_enum_items_add_value(&item, &totitem, Obuttons_context_items,
OBJCONTEXT_TUTORIAL);
+
+	RNA_enum_item_end(&item, &totitem);
+	*r_free = true;
+
+	return item;
+}
+
+static EnumPropertyItem *rna_SpaceTutorial_texture_context_itemf(bContext *C, PointerRNA
```

```
*UNUSED(ptr),
+	PropertyRNA *UNUSED(prop), bool *r_free)
+{
+	EnumPropertyItem *item = NULL;
+	int totitem = 0;
+
+	if (ED_texture_context_check_world(C)) {
+		RNA_enum_items_add_value(&item, &totitem, buttons_texture_context_items,
SB_TEXC_WORLD);
+	}
+
+	if (ED_texture_context_check_lamp(C)) {
+		RNA_enum_items_add_value(&item, &totitem, buttons_texture_context_items,
SB_TEXC_LAMP);
+	}
+	else if (ED_texture_context_check_material(C)) {
+		RNA_enum_items_add_value(&item, &totitem, buttons_texture_context_items,
SB_TEXC_MATERIAL);
+	}
+
+	if (ED_texture_context_check_particles(C)) {
+		RNA_enum_items_add_value(&item, &totitem, buttons_texture_context_items,
SB_TEXC_PARTICLES);
+	}
+
+	if (ED_texture_context_check_linestyle(C)) {
+		RNA_enum_items_add_value(&item, &totitem, buttons_texture_context_items,
SB_TEXC_LINESTYLE);
+	}
+
+	if (ED_texture_context_check_others(C)) {
+		RNA_enum_items_add_value(&item, &totitem, buttons_texture_context_items,
SB_TEXC_OTHER);
+	}
+
+	RNA_enum_item_end(&item, &totitem);
+	*r_free = true;
+
+	return item;
+}
+
+static void rna_SpaceTutorial_texture_context_set(PointerRNA *ptr, int value)
+{
+	SpaceTutorial *stutorial = (SpaceTutorial *)(ptr->data);
+
+	/* User action, no need to keep "better" value in prev here! */
+	stutorial->texture_context = stutorial->texture_context_prev = value;
+}
+
+/*-----------------------------------------------------------------------------------------------*/

 /* Space Properties */

 /* note: this function exists only to avoid id refcounting */
```

@@ -3342,30 +3590,129 @@

```
	RNA_api_space_text(srna);
 }

-// bfa - our new tutorial editor
++// bfa - our new tutorial editor
++//static void rna_def_space_tutorial(BlenderRNA *brna)
++//{
++//	StructRNA *srna;
++//	//PropertyRNA *prop;
++//
++//	srna = RNA_def_struct(brna, "SpaceTutorialEditor", "Space");
++//	RNA_def_struct_sdna(srna, "SpaceTutorial");
++//	RNA_def_struct_ui_text(srna, "Space Tutorial Editor", "Tutorial editor space data");
++//
++//}
++
++/*-----------------------------------------------------------------------------------------------------*/
++
++//bfa - our new tutorial editor
++
 static void rna_def_space_tutorial(BlenderRNA *brna)
 {
	StructRNA *srna;
-	//PropertyRNA *prop;
-
-	srna = RNA_def_struct(brna, "SpaceTutorialEditor", "Space");
++	PropertyRNA *prop;
++
++	static EnumPropertyItem align_items[] = {
++		{ TUTORIAL_HORIZONTAL, "HORIZONTAL", 0, "Horizontal", "" },
++		{ TUTORIAL_VERTICAL, "VERTICAL", 0, "Vertical", "" },
++		{ 0, NULL, 0, NULL, NULL }
++	};
++
++	srna = RNA_def_struct(brna, "SpaceTutorial", "Space");
	RNA_def_struct_sdna(srna, "SpaceTutorial");
-	RNA_def_struct_ui_text(srna, "Space Tutorial Editor", "Tutorial editor space data");
-
++	RNA_def_struct_ui_text(srna, "Space Tutorial", "tutorial space data");
++
++	prop = RNA_def_property(srna, "context", PROP_ENUM, PROP_NONE);
++	RNA_def_property_enum_sdna(prop, NULL, "mainb");
++	RNA_def_property_enum_items(prop, buttons_context_items);
++	RNA_def_property_enum_funcs(prop, NULL, "rna_SpaceTutorial_context_set",
"rna_SpaceTutorial_context_itemf");
++	RNA_def_property_ui_text(prop, "Context", "Type of active data to display and edit");
++	RNA_def_property_update(prop, NC_SPACE | ND_SPACE_TUTORIAL, NULL);
++
++	prop = RNA_def_property(srna, "align", PROP_ENUM, PROP_NONE);
++	RNA_def_property_enum_sdna(prop, NULL, "align");
++	RNA_def_property_enum_items(prop, align_items);
++	RNA_def_property_enum_funcs(prop, NULL, "rna_SpaceTutorial_align_set", NULL);
++	RNA_def_property_ui_text(prop, "Align", "Arrangement of the panels");
++	RNA_def_property_update(prop, NC_SPACE | ND_SPACE_TUTORIAL, NULL);
++
```

```
+	/* pinned data */
+	prop = RNA_def_property(srna, "pin_id", PROP_POINTER, PROP_NONE);
+	RNA_def_property_pointer_sdna(prop, NULL, "pinid");
+	RNA_def_property_struct_type(prop, "ID");
+	/* note: custom set function is ONLY to avoid rna setting a user for this. */
+	RNA_def_property_pointer_funcs(prop, NULL, "rna_SpaceTutorial_pin_id_set",
+		"rna_SpaceTutorial_pin_id_typef", NULL);
+	RNA_def_property_flag(prop, PROP_EDITABLE | PROP_NEVER_UNLINK);
+	RNA_def_property_update(prop, NC_SPACE | ND_SPACE_TUTORIAL,
"rna_SpaceTutorial_pin_id_update");
+
+	prop = RNA_def_property(srna, "use_pin_id", PROP_BOOLEAN, PROP_NONE);
+	RNA_def_property_boolean_sdna(prop, NULL, "flag", TUTORIAL_PIN_CONTEXT);
+	RNA_def_property_ui_text(prop, "Pin ID", "Use the pinned context");
+
 }
+/*-----------------------------------------------------------------------------------------------------*/

 static void rna_def_space_dopesheet(BlenderRNA *brna)
 {
```

source/blender/makesrna/intern/rna_userdef.c

@@ -1897,6 +1897,56 @@

```
	RNA_def_property_update(prop, 0, "rna_userdef_update");
 }

+/* bfa - space_tutorial */
+static void rna_def_userdef_theme_space_tutorial(BlenderRNA *brna)
+{
+	StructRNA *srna;
+	PropertyRNA *prop;
+
+	srna = RNA_def_struct(brna, "ThemeTutorialEditor", NULL);
+	RNA_def_struct_sdna(srna, "ThemeSpace");
+	RNA_def_struct_clear_flag(srna, STRUCT_UNDO);
+	RNA_def_struct_ui_text(srna, "Theme tutorial", "Theme settings for the tutorial");
+
+	rna_def_userdef_theme_spaces_main(srna);
+
+	prop = RNA_def_property(srna, "match", PROP_FLOAT, PROP_COLOR_GAMMA);
+	RNA_def_property_array(prop, 3);
+	RNA_def_property_ui_text(prop, "Filter Match", "");
+	RNA_def_property_update(prop, 0, "rna_userdef_update");
+
+	prop = RNA_def_property(srna, "selected_highlight", PROP_FLOAT,
PROP_COLOR_GAMMA);
+	RNA_def_property_array(prop, 3);
+	RNA_def_property_ui_text(prop, "Selected Highlight", "");
+	RNA_def_property_update(prop, 0, "rna_userdef_update");
+}
+
+
```

```
static void rna_def_userdef_theme_space_outliner(BlenderRNA *brna)
{
	StructRNA *srna;
```

@@ -3033,8 +3083,8 @@

```
		{16, "FILE_BROWSER", ICON_FILESEL, "File Browser", ""},
		{17, "CONSOLE", ICON_CONSOLE, "Python Console", ""},
		{20, "CLIP_EDITOR", ICON_CLIP, "Movie Clip Editor", ""},
-		{ 21, "TUTORIAL_EDITOR", ICON_OBJECTINFO, "Tutorial Editor", "" }, // bfa - our new tutorial editor
+		{21, "TUTORIAL_EDITOR", ICON_OBJECTINFO, "Tutorial Editor", "" }, // bfa - our new tutorial editor

		{0, NULL, 0, NULL, NULL}
	};
```

@@ -3169,19 +3219,19 @@

```
	RNA_def_property_struct_type(prop, "ThemeClipEditor");
	RNA_def_property_ui_text(prop, "Clip Editor", "");

+	/*bfa - tutorial editor */

-	//prop = RNA_def_property(srna, "tutorial_editor", PROP_POINTER, PROP_NONE);
-	//RNA_def_property_flag(prop, PROP_NEVER_NULL);
-	//RNA_def_property_pointer_sdna(prop, NULL, "ttutorial");
-	//RNA_def_property_struct_type(prop, "ThemeTutorialEditor");
-	//RNA_def_property_ui_text(prop, "Tutorial Editor", "");
-

+	prop = RNA_def_property(srna, "tutorial_editor", PROP_POINTER, PROP_NONE);
+	RNA_def_property_flag(prop, PROP_NEVER_NULL);
+	RNA_def_property_pointer_sdna(prop, NULL, "ttutorial");
+	RNA_def_property_struct_type(prop, "ThemeTutorialEditor");
+	RNA_def_property_ui_text(prop, "Tutorial Editor", "");
}

static void rna_def_userdef_addon(BlenderRNA *brna)
```

@@ -3274,6 +3324,8 @@

```
	rna_def_userdef_theme_space_clip(brna);
	rna_def_userdef_theme_colorset(brna);
	rna_def_userdef_themes(brna);
+	rna_def_userdef_theme_space_tutorial(brna);
}

static void rna_def_userdef_solidlight(BlenderRNA *brna)
```

source/blender/makesrna/RNA_access.h

@@ -576,8 +576,8 @@
Fix for wrong RNA name

```
 extern StructRNA RNA_SpaceUserPreferences;
 extern StructRNA RNA_SpaceView3D;
 extern StructRNA RNA_SpaceClipEditor;
-extern StructRNA RNA_SpaceTutorialEditor;
+extern StructRNA RNA_ThemeTutorialEditor;
 extern StructRNA RNA_Speaker;
 extern StructRNA RNA_SpeedControlSequence;
 extern StructRNA RNA_Spline;
```

@@ -640,6 +640,8 @@

```
 extern StructRNA RNA_ThemeOutliner;
 extern StructRNA RNA_ThemeProperties;
 extern StructRNA RNA_ThemeSequenceEditor;
+extern StructRNA RNA_ThemeTutorialEditor;
 extern StructRNA RNA_TextSequence;
 extern StructRNA RNA_ThemeSpaceGeneric;
 extern StructRNA RNA_ThemeSpaceGradient;
```

source/blender/python/intern/bpy_rna_callback.c

@@ -174,8 +174,8 @@
Fix for wrong RNA name

```
        if (srna == &RNA_SpaceConsole)          return SPACE_CONSOLE;
        if (srna == &RNA_SpaceUserPreferences)  return SPACE_USERPREF;
        if (srna == &RNA_SpaceClipEditor)       return SPACE_CLIP;
-       if (srna == &RNA_SpaceTutorialEditor)   return SPACE_TUTORIAL; // bfa - second part tutorial editor
+       if (srna == &RNA_ThemeTutorialEditor)   return SPACE_TUTORIAL; // bfa - second part tutorial editor

        return -1;
 }
```

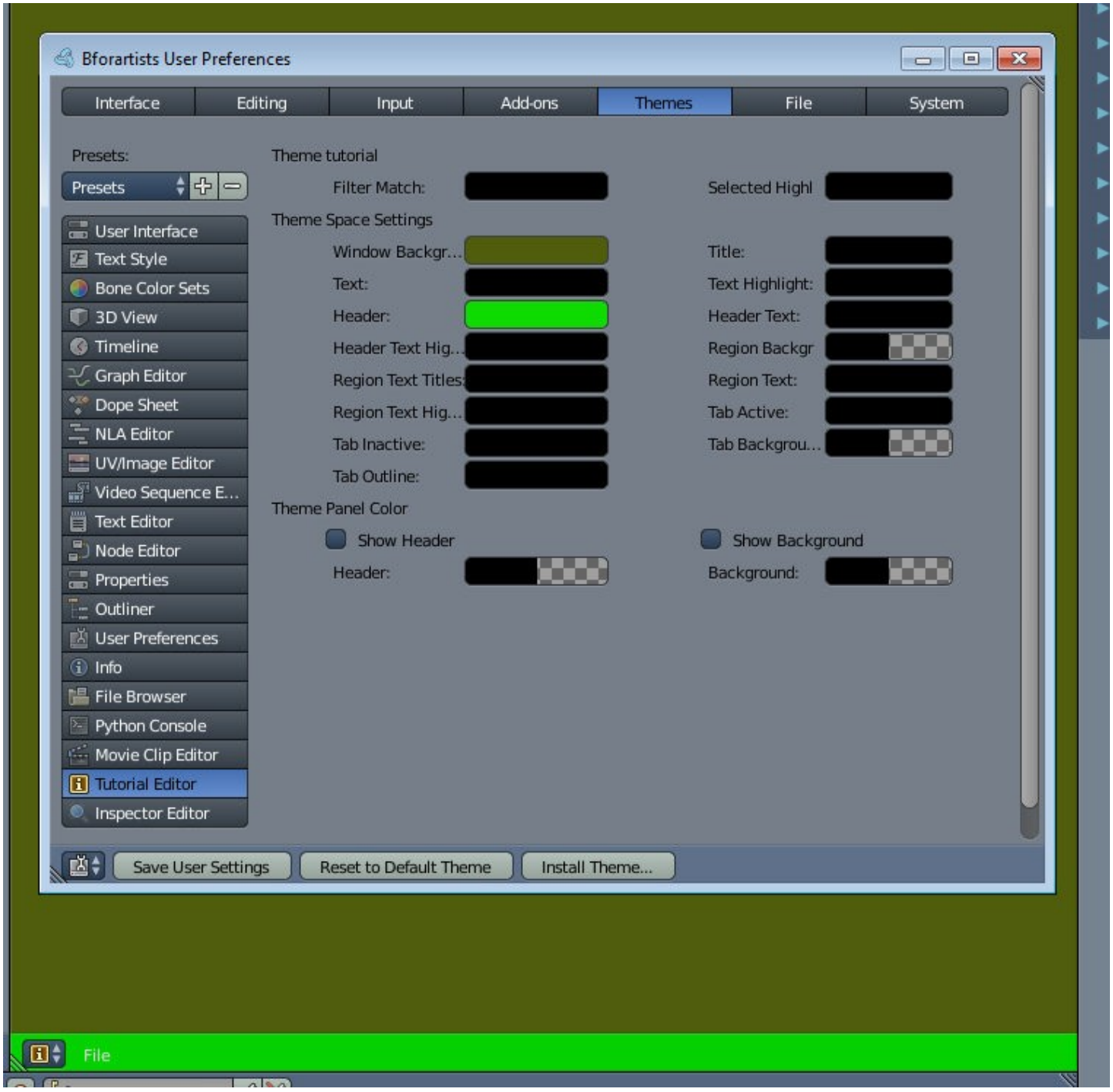
source/blender/windowmanager/intern/wm_keymap.c

@@ -1670,7 +1670,16 @@

```
        {
                km = WM_keymap_find_all(C, "Window", 0, 0);
        }
-
+
+       /* bfa - Tutorial editor */
+       else if (STRPREFIX(opname, "TUTORIAL_OT")) {
+               km = WM_keymap_find_all(C, "Tutorial", sl->spacetype, 0);
+       }
+
```

```
+       /* bfa - Inspector editor */
+       else if (STRPREFIX(opname, "INSPECTOR_OT")) {
+               km = WM_keymap_find_all(C, "Inspector", sl->spacetype, 0);
+       }

        /* 3D View */
        else if (STRPREFIX(opname, "VIEW3D_OT")) {
```

And when everything works as thought then we should have this now:



Content

The content was the part that i never got working. I got a ton of errors with the files used from somebody else. And the person was not to convince to give me the whole project file where the content part supposedly worked.

So there was no chance to fix the problem without to rewrite it by myself. And at that point i abandoned the task to implement a new editor type.