
sqf

content

introDuction

digital picture browsing - spectacular scenic view

architektur wurde und wird in der hauptsache über bilder vermittelt. im diplomwahlfach replay bild-o-mat entwickeln wir einen digitalen bildbrowser als arbeits- und präsentationswerkzeug für heutige architektInnen.

hintergrund dieser praktischen arbeit bietet die theoretische auseinandersetzung mit dem bildgebrauch in der architektur: wichtig zum verständnis sind zwei mediale zäsuren: zum einen hat das aufkommen der fotografie im 19. jh. die massenhafte verbreitungsmöglichkeit und individuelle verfügbarkeit von bildern ermöglicht. zum anderen erlaubt es die verbreitung der digitalen medien eine sehr große anzahl von bildern herzustellen, zu speichern und zu verwalten.

bei replay bild-o-mat untersuchen wir die beschaffenheit des digitalen bildes, kompressionsverfahren und annotationsmöglichkeiten sowie eine reihe von werkzeugen, mit denen man bildsammlungen strukturieren und zugänglich machen kann. beispiele sind professionelle bilddatenbanken sowie künstlerische projekte. auf der grundlage dieser recherche werden verschiedene bildbrowser in macromedia flash erstellt. diese bildbrowser werden auf der grundlage der easyDB datenbank-oberflaeche (forschungsdatenbank der professur für CAAD) zugreifen.

veranstaltungstermin und -ort:
donnerstags 14 bis 17 Uhr, HIL E 65
zweiwöchiger workshop in den semesterferien

leitung und kontakt:
katharina bosch | kai rüdenauer | susanne schumacher | torsten spindler |

introduction - sqf s ° q ° f stylequalityform *page*

concept | **phase1**

der s ° q ° f browser ist als ein subjektiver, dem persoenlichen empfinden sich anpassender bildbrowser konzipiert. er soll eine art impulsgeber im entwurfsprozess sein, bei dem es sich nicht um die wissenschaftliche auseinandersetzung, sondern um das sonst uebliche schnelle blaettern in zeitschriften und architekturbuechern handeln wird.
einer anfaenglichen, konventionellen suche (search page) nach kriterien wie author, stichwoerter, stadt, quelle bzw. sonstiger metadaten. folgt der wesentliche teil.
das suchergebnis wird durch ein zweites bild, sowie durch alle, der suche entsprechenden bilder in thumbnailgroeße, ergaenzt. den beiden großen suchergebnissen sind je drei, anfaenglich zufaellige, subjektive, sowie oberflaechliche werte zugeordnet:
style, quality, form.
bei diesen werten geht es nicht um eine fundierte kritische betrachtung des abgebideten werks, sondern um eine rein emotionale und schnelle entscheidung, „gut“ oder „schlecht“, wobei dies in werte von null bis zehn differenziert (definiert) wird.
style beschreibt hier die parameter der „coolness“, sowohl des fotos, als auch des abgebildeten inhalts. dazu zaehlen z.B. farbgebung, oder formale architektursprache, wie „blob“ oder hadid´sche winkel.
quality versucht der ansonsten oberflaechlichen betrachtungsweise etwas ernsthaftigkeit entgegen zu stellen. ordnet man ein bild in der style-kategorie z.B. sehr negativ ein, so kann das abgebildete natuerlich trotzdem von hoher qualitaet sein.
form beschreibt die praegnanz der koerperhaftigkeit des abgebildeten. hier reicht die skala von 0 wie banal bis 10 wie objekthaft.

der nutzer des browsers wird dazu aufgefordert, die anfaenglich zufaellig gesetzten werte, seinem empfinden nach zu korrigieren. nach und nach filtert sich die bildermenge nach oben genannten kriterien in gut bis schlecht (kriterienspektrum 0-10).
ab diesem moment funktioniert der wesentliche aspekt des s ° q ° f konzepts:

kontraste
jeder wertung folgt die automatische zuordnung eines bildes mit dem gegenwert.
bewertet man eines der beiden großen bilder z.B. mit dem form-wert 10, so wird auf der anderen seite ein bild mit der wertung 0 angezeigt. es entstehen die jeweils größtmöglichen kontraste, die die bilder befruchten, wie hell-dunkel-, oder komplementärkontraste in der kunst. sehgewohnheiten, die sich beim üblichen ansehen von fachzeitschriften, oder architekturmonographien, etc. gebildet haben, werden gebrochen.

ergebnisse results
auf einer zweiten ebene (result page) des s ° q ° f kann man sich die gesamten datenbankbestände nach entsprechender wertung sortiert ansehen.
hier schiebt man den regler in der jeweiligen kategorie einfach auf den gewuenschten wert und erhaelt so einen ueberblick bezueglich inhalt der datenbank und fremdbewertungen.

concept | **phase2** (in progress)

averagefunktion
als erweiterung des s ° q ° f konzepts ist eine averagefunktion vorgesehen, diese ist in der easyDB bereits angelegt.
neben der subjektiven wertung wird hier, entsprechend des vorhandenen results, der durchschnittsgeschmack ausgegeben.
in der averagefunktion werden die einzelnen bewertungen summiert und auf einer onRelease eingabe als durchschnittswerte ermittelt und angezeigt.

concept | **phase3** (in progress)

uploadfunktion
ergaenzend soll es dem user moeglich sein, die datenbank durch eigene photos zu ergaenzen. dies ist bereits ueber die easyDB moeglich.

concept, visualize & programming:
jenny weiss | ferdinand kersten | stephan albrecht |

*01. **link: sqf-bildbrowser***

01. oeffnet bildbrowser

02. oeffnet page zum neusten flas-player

03. zeigt aktuelles ausgewaehltes bild in einem groesseren fenster

04. jeweiliger gegensatz des aktiven bildes

05. zeigt dateiinformationen zum aktiven bild

06. anzahl der gefundenen treffer aufgrund einer auswahl-eingabe

07. durch switchen auf eine ander

08. auswahl wird jeweils mit den ersten sieben bildern in die leiste geladen

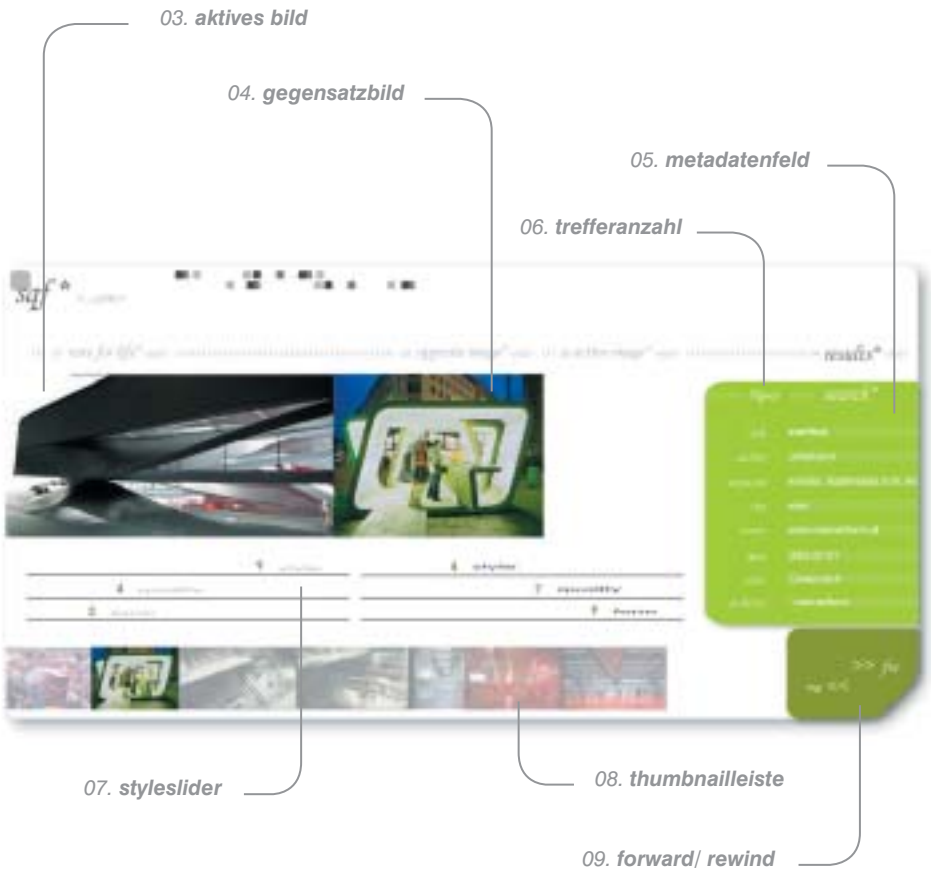
09. die jeweils naechsten sieben bilder der auswahl werden geladen

10. alle bilder einer stylewert-kategorie werden in ein dynamisches bildband geladen

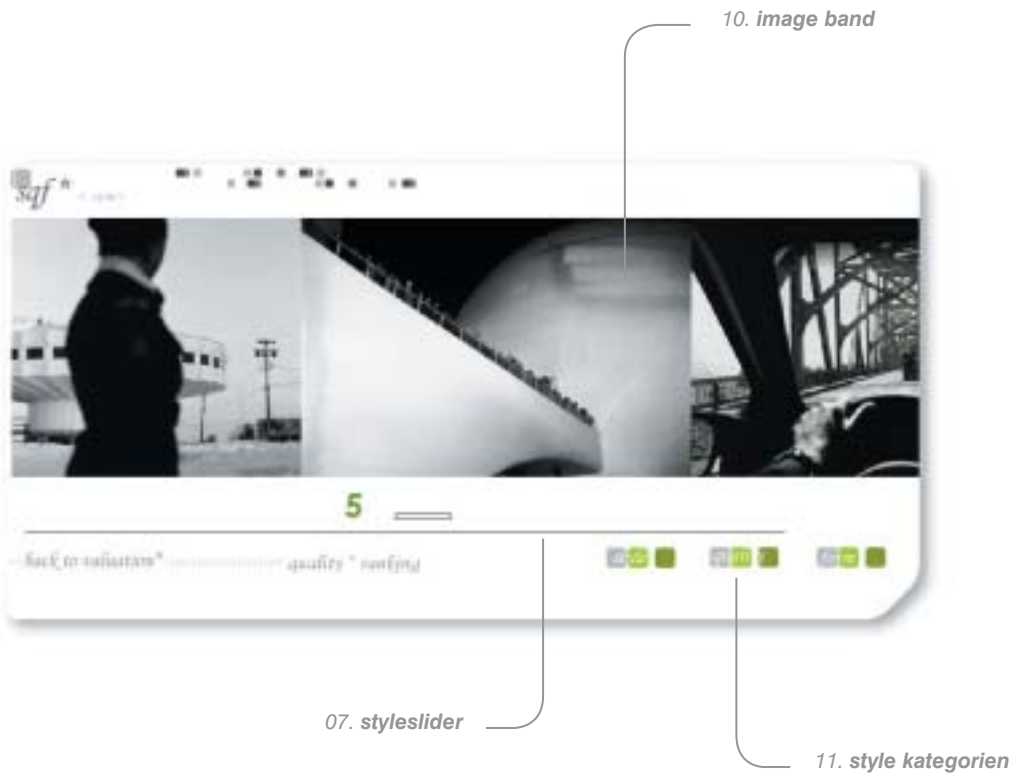
11. durch switchen auf eine andere stylewert-kategorie wird die jeweilige auswahl an bildern in das image band geladen

*02. link: **macromedia-page***

start-page | **ebene 01**



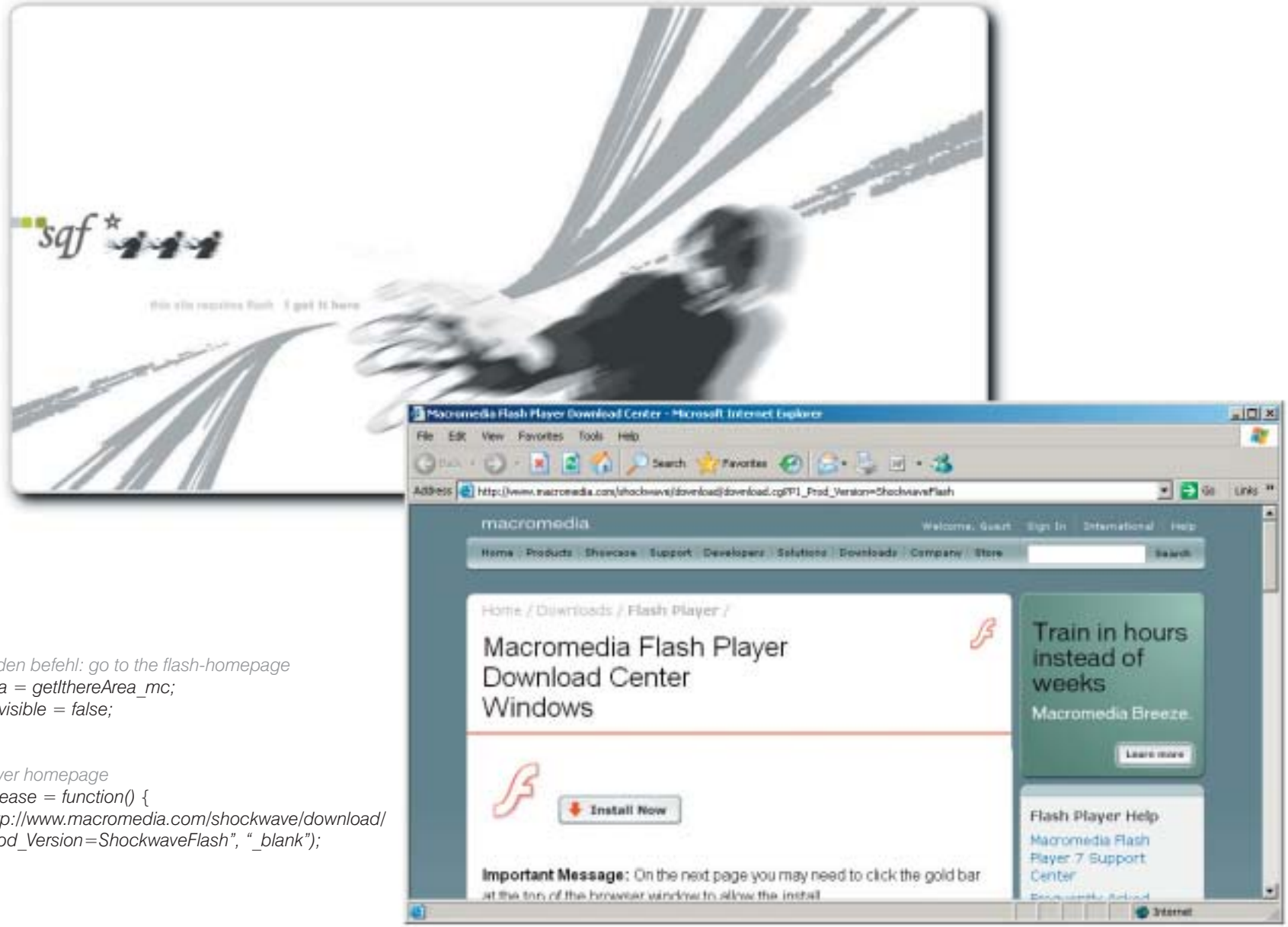
search-page | **ebene 02**



result-page | **ebene 03**

realiSation

| ... die start-page (ebene 01) beinhaltet einerseits eine einfuehrende grafik, als auch die verlinkung zu den 2 hauptseiten (search & result-page), auf der die eigentliche datenbank abfrage abgerufen wird. ebenso wurde ein link zur macromedia-page eingebaut, wo man sich den neuesten flash-player herunterladen kann, der fuer die sqf page benoetigt wird ... |




```
// setze hitArea fuer den befehl: go to the flash-homepage
getIthere_mc.hitArea = getIthereArea_mc;
getIthereArea_mc._visible = false;


// gehe zur flash-player homepage
getIthere_mc.onRelease = function() {
        getURL("http://www.macromedia.com/shockwave/download/
download.cgi?P1_Prod_Version=ShockwaveFlash", "_blank");
};
```

| ... auf der aufgerufenen search-page (ebene 02) ist die auswahl (ohne eingabe) auf null gesetzt, d.h. man wird dazu aufgefordert eine eigene auswahl ueber das metadatenfeld oder die styleslider aufzurufen ... |

```
// laedt die entsprechenden metadaten in die textfelder.
function metaupdate(index) {
	_root.myArg0_txt.text = _global.imageControl_array[index].titel;
	_root.myArg1_txt.text = _global.imageControl_array[index].herkunft;
	_root.myArg2_txt.text = _global.imageControl_array[index].werkzusammenhang;
	_root.myArg3_txt.text = _global.imageControl_array[index].masse;
	_root.myArg4_txt.text = _global.imageControl_array[index].bildnachweis;
	_root.myArg5_txt.text = _global.imageControl_array[index].datierung;
	_root.myArg6_txt.text = _global.imageControl_array[index].erlaeuterungen;
	_root.myArg7_txt.text = String(_global.imageControl_array[index].myReplay.Architekt_vorn)+String(" ")+String(_global.imageControl_array[index].myReplay.Architekt_nach);
}
```

| ... mit einer eingabe und dem druecken auf search, wird die datenbank nun auf die entsprechenden eingebenden werte durchsucht, und die gefundenen images als vorschaubilder in die thumbnailleiste hochgeladen. die ersten sieben bilder werden nun in der thumbleiste angezeigt. durch eine forward- bzw. rewind eingabe koennen die jeweils naechsten bzw. letzten sieben bilder angezeigt werden. ein zaehler neben dem search button zeigt die aktuelle anzahl gefundener images der jeweiligen auswahl an. ausserdem werden zwei zufaellig gewaehlte images aus der suchanfrage mittig auf die page, mit hoeherer aufloesung, ausgegeben ... |

```
// loescht textfelder bie neuer suchanfrage, setzt zufallsvariable.
textkill = 1;
scale = 3/4;
meineWertVariable = "Sty_Wert";
var lastclicked;
function randRange(min:Number, max:Number):Number {
        var randomNum:Number = Math.round(Math.random()*(max-min))+min;
        return randomNum;
}
```


```
// positionierung der beiden bildfenster der search-page
_root.stephan_mc._x = 30;
_root.stephan_mc._y = 59.1;
_root.peter_mc.createEmptyMovieClip("image", 1);
_root.stephan_mc.createEmptyMovieClip("image", 1);
_root.peter_mc.onEnterFrame = function() {
        this._x = _root.stephan_mc._x+_root.stephan_mc._width;
        this._y = _root.stephan_mc._y;
};
```

| ... die hochgeladenen thumbnails koennen individuel per mausklick als einer der zwei images (aktives bild & gegensatzbild) mit hoeherer aufloesung angezeigt werden. automatisch werden zur anzeige des neuen, groesseren images die metadaten ausgetauscht und aktualisiert ... |


```
// reagieren der thumbnailleiste auf onRelease & onRollOver
                    _root.ferdiParent_mc["ferdi_mc"+i].onRelease = function() {
                            if (lastclicked == 0) {
                                    _root.setBigImage2(this.myPosInArray,false);
                            } else {
                                    _root.setBigImage(this.myPosInArray);
                            }
                            arrangeImages(this.myPosInArray);
                    };


                    _root.ferdiParent_mc["ferdi_mc"+i].onRollOver = function() {
                            this._alpha = 100;
                    };
                    _root.ferdiParent_mc["ferdi_mc"+i].onRollOut = function() {
                            this._alpha = 40;
                    };
            }
};
```

| ... die eigentliche konzept-idee besteht darin, dass man styleslider (3 kategorien) zum beurteilen der images hat, um ihnen individuell und subjektiv bewertungen zu zuweisen. diese werden nicht als durchschnittswerte ausgegeben, sondern direkt in die datenbank geschrieben und dort gespeichert. sie sollen somit den naechsten besucher dazu anregen die bewertung zu beurteilen, indem er sie wieder ueberschreiben kann. dadurch wird garantiert, dass sich die datenbank staendig veraendert, dadurch provozierender wirkt, und nicht durch durchschnittswerte in ihrer subjektiv angedachten idee verwischt wird ... |

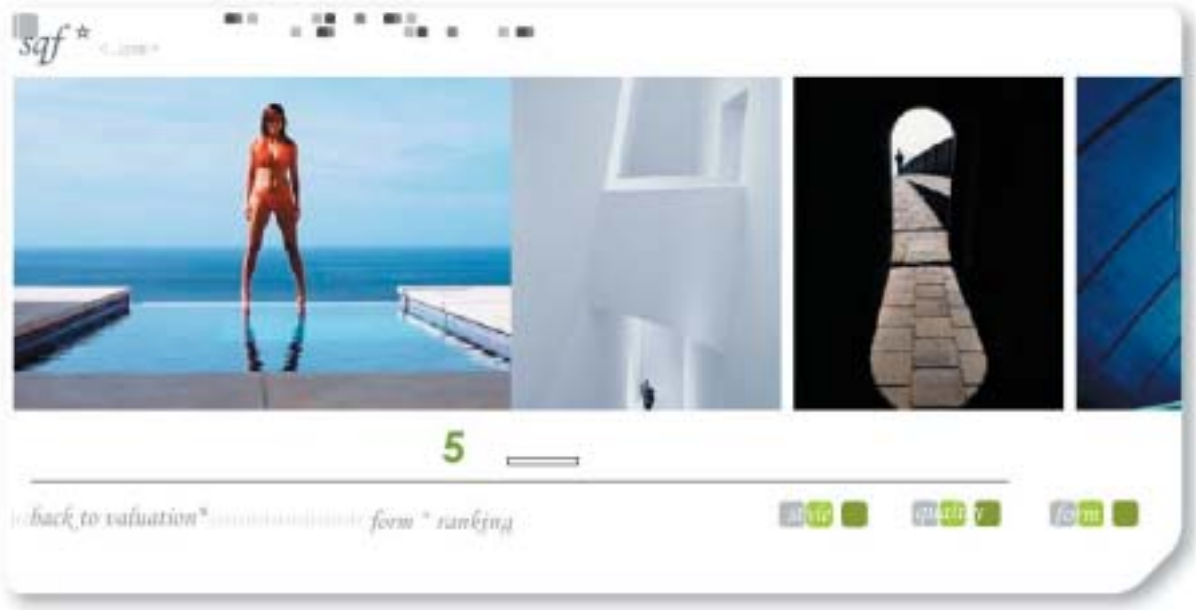
```
// reagieren des linken bildcontainers auf anklicken eines thumbnails, eines stylesliders.
setBigImage = function (index) {
        textkill = 0;
        _root.stephan_mc.myindex = index;
        _root.stephan_mc._x = 50;
        _root.stephan_mc._y = 170;
        _root.stephan_mc.image.loadMovie(_global.imageControl_array[index].urlMed, 1);
        stephanForm_mc.slide_mc._x = _global.imageControl_array[index].myReplay.Form_Wert[0]*10;
        stephanQuality_mc.slide_mc._x = _global.imageControl_array[index].myReplay.Qu_Wert[0]*10;
        stephanStyle_mc.slide_mc._x = _global.imageControl_array[index].myReplay.Sty_Wert[0]*10;
        if ((_global.imageControl_array[index].orig_height/_global.imageControl_array[index].orig_width)<1) {
                _root.stephan_mc._yscale = 50*_global.imageControl_array[index].orig_width/_global.imageControl_array[index].orig_height;
                _root.stephan_mc._xscale = 50*_global.imageControl_array[index].orig_width/_global.imageControl_array[index].orig_height;
        } else {
                _root.stephan_mc._yscale = 50;
                _root.stephan_mc._xscale = 50;
        }
        metaupdate(index);
};
```

| ... will man sich nun alle bilder einer bestimmten bewertung anzeigen lassen, kann man sich ueber eine result eingabe alle gefundenen images zu einem speziellen wert der jeweiligen styleslider hochaufloesender version hochladen lassen. diese auswahl wird in ein dynamisch ablaufendes image band auf die result-page (ebene 03) geladen ... |


```
// funktion fuer die result page, laedt extern bilder in den “monsterarray”, schaltet thumbnailleiste auf unsichtbar.
function result_func(randNum) {
        trace(“----------------RESULT FUNCTION: “);
        _root.ferdiParent_mc._visible = 0;

        big_ranking_mc.slide_mc._x = randNum*10;
        big_ranking_mc._visible = 0;
        delete _root.monsterarray;
        _root.monsterarray = new Array();
        _root.askEasyDB(“erlaeuterungen”, “<”+meineWertVariable+”>” + randNum +”</”+meineWertVariable+”>”,
“normal”, _root.monsterarray, _root.aufbau_ost, “normal”);
        }
```

| ... auf der result-page (ebene 03) kann man sich nun fuer jeden bestimmten style-wert bzw. jede style-kategorie alle zu findenden images anzeigen lassen. durch zurueckspringen auf die search-page (ebene 02, back to valuation), und das eingeben neuer werte kann die auswahl der result-page nun in sekunden (auf wunsch komplett) veraendert werden, und somit aktiv beeinflusst werden. das navigieren der images (result-page) erfolgt ueber die mausposition, und ist somit auf basis einer onRollOver anweisung staendig in bewegung ... |

```
// funktion fuer die result page, laedt die bilder auf die page.
function aufbau_ost() {
	trace("----------------AUFBAU OST FUNCTION: ");
	trace("----------------MONSTERARRAY: "+ _root.monsterarray.length);
	big_ranking_mc._visible = 1;
	bilderplot();
}
function bilderplot() {
	for(obj in _root.all_mc){
		_root.all_mc[obj].removeMovieClip();
	}
	_root.all_mc.removeMovieClip();
	trace("---------------------------- before break ");
	trace("---------------------------- after break ");
	trace("---------------------------- INSIDE BILDERPLOT ");

	style_bilder = _root.monsterarray;
	b_index = 0;
	if (style_bilder.length>100) {
		max = 100;
	} else {
		max = style_bilder.length;
	}
	trace("----------------------> gefunden: "+style_bilder.length);
	trace("----------------------> gefunden: "+_root.monsterarray.length);
	_root.gesamtbreite = 0;

	// erstellt den clip "all_mc", setzt maske hierfuer fest, gibt dem slider seine bewegungsparameter.
	_root.createEmptyMovieClip("all_mc", _root.getNextHighestDepth());
		_root.all_mc.onEnterFrame = function() {
			_root.all_mc.setMask(_root.mask_mc);
		if (_ymouse>_root.all_mc._y) {
			if (_ymouse<_root.all_mc._y+240) {
				if (_root._xmouse < Stage.width/2) {
					_root.all_mc._x += 20;
				}
				else{
					_root.all_mc._x -= 20;
				}

			}
		}
	};
```




actionScript

```
stop();

// bindet externes xml in die page ein.
#include “connectEasyDB_v1.0.as”

// loescht textfelder bei neuer suchanfrage, setzt zufallsvariable.
textkill = 1;
scale = 3/4;
meineWertVariable = “Sty_Wert”;
var lastclicked;
function randRange(min:Number, max:Number):Number {
	var randomNum:Number = Math.round(Math.random()*(max-min))+min;
	return randomNum;
}

// funktion fuer die result-page (ebene 03), laedt extern bilder in den “monsterarray”, schaltet thumbnailleiste auf unsichtbar.
function result_func(randNum) {
	trace(“----------RESULT FUNCTION: “);
	_root.ferdiParent_mc._visible = 0;
	big_ranking_mc.slide_mc._x = randNum*10;
	big_ranking_mc._visible = 0;
	delete _root.monsterarray;
	_root.monsterarray = new Array();
	_root.askEasyDB(“erlaeuterungen”, “<”+meineWertVariable+”>” + randNum +”</”+meineWertVariable+”>”, “normal”, _root.monsterarray, _root.aufbau_ost, “normal”);
	}

// funktion fuer die result-page (ebene 03), laedt die bilder auf die page.
function aufbau_ost() {
	big_ranking_mc._visible = 1;
	bilderplot();
}
function bilderplot() {
	for(obj in _root.all_mc){
		_root.all_mc[obj].removeMovieClip();
	}
	_root.all_mc.removeMovieClip();

	style_bilder = _root.monsterarray;
	b_index = 0;
	if (style_bilder.length>100) {
		max = 100;
	}
```

```
else {
		max = style_bilder.length;
}
trace("----------------------> gefunden: "+_root.monsterarray.length);
_root.gesamtbreite = 0;

// erstellt den clip "all_mc", setzt maske hierfuer fest, gibt dem slider seine bewegungsparameter.
_root.createEmptyMovieClip("all_mc", _root.getNextHighestDepth());
		_root.all_mc.onEnterFrame = function() {
			_root.all_mc.setMask(_root.mask_mc);
		if (_ymouse>_root.all_mc._y) {
			if (_ymouse<_root.all_mc._y+240) {
				if (_root._xmouse < Stage.width/2) {
					_root.all_mc._x += 20;
				}
				else{
					_root.all_mc._x -= 20;
				}
			}
		}
};


// erstellt die clips "result_mc" + i, skaliert sie.
for (i=0; i<max; i++) {
		kuerzel = _root.all_mc.createEmptyMovieClip("result_mc"+i, _root.all_mc.getNextHighestDepth());
		kuerzel.createEmptyMovieClip("image", 1);
		kuerzel.image.loadMovie(style_bilder[i].urlMed, 1);
		trace("loadMovie");
		if ((style_bilder[i].orig_height/style_bilder[i].orig_width)<1) {
			kuerzel._yscale = 90*style_bilder[i].orig_width/style_bilder[i].orig_height;
			kuerzel._xscale = 90*style_bilder[i].orig_width/style_bilder[i].orig_height;
		} else {
			kuerzel._yscale = 85;
			kuerzel._xscale = 85;
}
		if (i>0) {
			var lastImage, lastBreitehöhe, currentImage;
			currentImage = _root.all_mc["result_mc"+i];
			lastImage = _root.all_mc["result_mc"+[i-1]];
			lastBreitehöhe = style_bilder[i-1].width/style_bilder[i-1].height;
			currentImage._x = lastImage._x+320*lastBreitehöhe;
			_root.gesamtbreite = _root.gesamtbreite+320*lastBreitehöhe;
```

```
                } else {
                        _root.all_mc[“result_mc”+i]._x = 0;
                        _root.gesamtbreite = 320;
                }
        }
        trace(“breite “+_root.gesamtbreite);
}



        // laedt die entsprechenden metadaten in die textfelder des metadatenfeldes.
function metaupdate(index) {
        _root.myArg0_txt.text = _global.imageControl_array[index].titel;
        _root.myArg1_txt.text = _global.imageControl_array[index].herkunft;
        _root.myArg2_txt.text = _global.imageControl_array[index].werkzusammenhang;
        _root.myArg3_txt.text = _global.imageControl_array[index].masse;
        _root.myArg4_txt.text = _global.imageControl_array[index].bildnachweis;
        _root.myArg5_txt.text = _global.imageControl_array[index].datierung;
        _root.myArg6_txt.text = _global.imageControl_array[index].erlaeuterungen;
        _root.myArg7_txt.text = String(_global.imageControl_array[index].myReplay.Architekt_vorn)+String(“ “)+String(_global.imageControl_array[index].myReplay.Architekt_nach);
}

   // positionierung der beiden bildfenster auf die search-page (ebene 02).
_root.stephan_mc._x = 30;
_root.stephan_mc._y = 59.1;
_root.peter_mc.createEmptyMovieClip(“image”, 1);
_root.stephan_mc.createEmptyMovieClip(“image”, 1);
_root.peter_mc.onEnterFrame = function() {
        this._x = _root.stephan_mc._x+_root.stephan_mc._width;
        this._y = _root.stephan_mc._y;
};


   // setzt die reaktion der textfelder (metadatenfeld) auf neue eingaben fest.
setInputAktion();
//_root.myArg3_txt.column = “city”;
function setInputAktion() {
        for (i=0; i<8; i++) {
                _root[“myArg”+i+”_txt”].index = i;
                _root[“myArg”+i+”_txt”].onSetFocus = function() {
                        _root.actText = this;
                        if (textkill == 1) {
                                clearInput();
```

```
                        }
                        if (textkill == 0) {
                                textkill == 1;
                        }
                        this.onChanged = function() {
                                clearInput(this.index);
                        };
                };
        }
}
function clearInput(ind) {
        for (i=0; i<8; i++) {
                if (i == ind) {
                        continue;
                }
                _root["myArg"+i+"_txt"].text = "";
        }
}
myArg1_txt.onChanged = function() {
};


    // zuordnug der hochgeladenen bilder in die bildcontainer "stephan_mc" und "peter_mc", skalierung und platzierung.
firstIndex = 0;
actionOnAnswer_xmlLoad = function () {
        _root.setBigImage(_root.randRange(0, _global.imageControl_array.length-1));
        _root.setBigImage2(_root.randRange(0, _global.imageControl_array.length-1));
        // treffer feld setzen
        _root.treffer_txt.text = _global.imageControl_array.length+"pics";
        // wo sollen die Bilder hin?
        _root.createEmptyMovieClip("ferdiParent_mc", _root.getNextHighestDepth());
        _root.ferdiParent_mc.setMask(_root.mask2_mc);
        _root.ferdiParent_mc._x = 50;
        _root.ferdiParent_mc._y = 465;
        _root.firstIndex = 0;
        for (i=0; i<7; i++) {
                // wo sollen die Bilder hingeladen werden
                _root.ferdiParent_mc.createEmptyMovieClip("ferdi_mc"+i, _root.ferdiParent_mc.getNextHighestDepth());
                _root.ferdiParent_mc["ferdi_mc"+i]._x = i*120;
                _root.ferdiParent_mc["ferdi_mc"+i]._alpha = 40;
                _root.ferdiParent_mc["ferdi_mc"+i].createEmptyMovieClip("image", 1);

                _root.ferdiParent_mc["ferdi_mc"+i].image.loadMovie(_global.imageControl_array[i].urlThumb, 1);
```

```
_root.ferdiParent_mc[“ferdi_mc”+i].myPosInArray = i;
_root.ferdiParent_mc[“ferdi_mc”+i].onRollOver = function() {

};
if ((_global.imageControl_array[i].orig_height/_global.imageControl_array[i].orig_width)<1) {
		_root.ferdiParent_mc[“ferdi_mc”+i]._yscale = 75*_global.imageControl_array[i].orig_width/_global.imageControl_array[i].orig_height;
		_root.ferdiParent_mc[“ferdi_mc”+i]._xscale = 75*_global.imageControl_array[i].orig_width/_global.imageControl_array[i].orig_height;
} else {
		_root.ferdiParent_mc[“ferdi_mc”+i]._yscale = 75;
		_root.ferdiParent_mc[“ferdi_mc”+i]._xscale = 75;
}

_root.ferdiParent_mc[“ferdi_mc”+0]._x = 0;
if (i>0) {
		var lastImage, lastBreitehöhe, currentImage;
		currentImage = _root.ferdiParent_mc[“ferdi_mc”+i];
		lastImage = _root.ferdiParent_mc[“ferdi_mc”+[i-1]];
		lastBreitehöhe = _global.imageControl_array[i-1].width/_global.imageControl_array[i-1].height;
		currentImage._x = lastImage._x+scale*100*lastBreitehöhe+0;
}

_root.peter_mc.onRelease = function() {
		lastclicked = 0;
		metaupdate(this.myindex);
};
_root.stephan_mc.onRelease = function() {
		metaupdate(this.myindex);
		lastclicked = 1;
};


// reagieren der thumbnailleiste auf onRelease & onRollOver
_root.ferdiParent_mc[“ferdi_mc”+i].onRelease = function() {
		if (lastclicked == 0) {
			_root.setBigImage2(this.myPosInArray,false);
		} else {
			_root.setBigImage(this.myPosInArray);
		}
		arrangeImages(this.myPosInArray);
};


_root.ferdiParent_mc[“ferdi_mc”+i].onRollOver = function() {
```

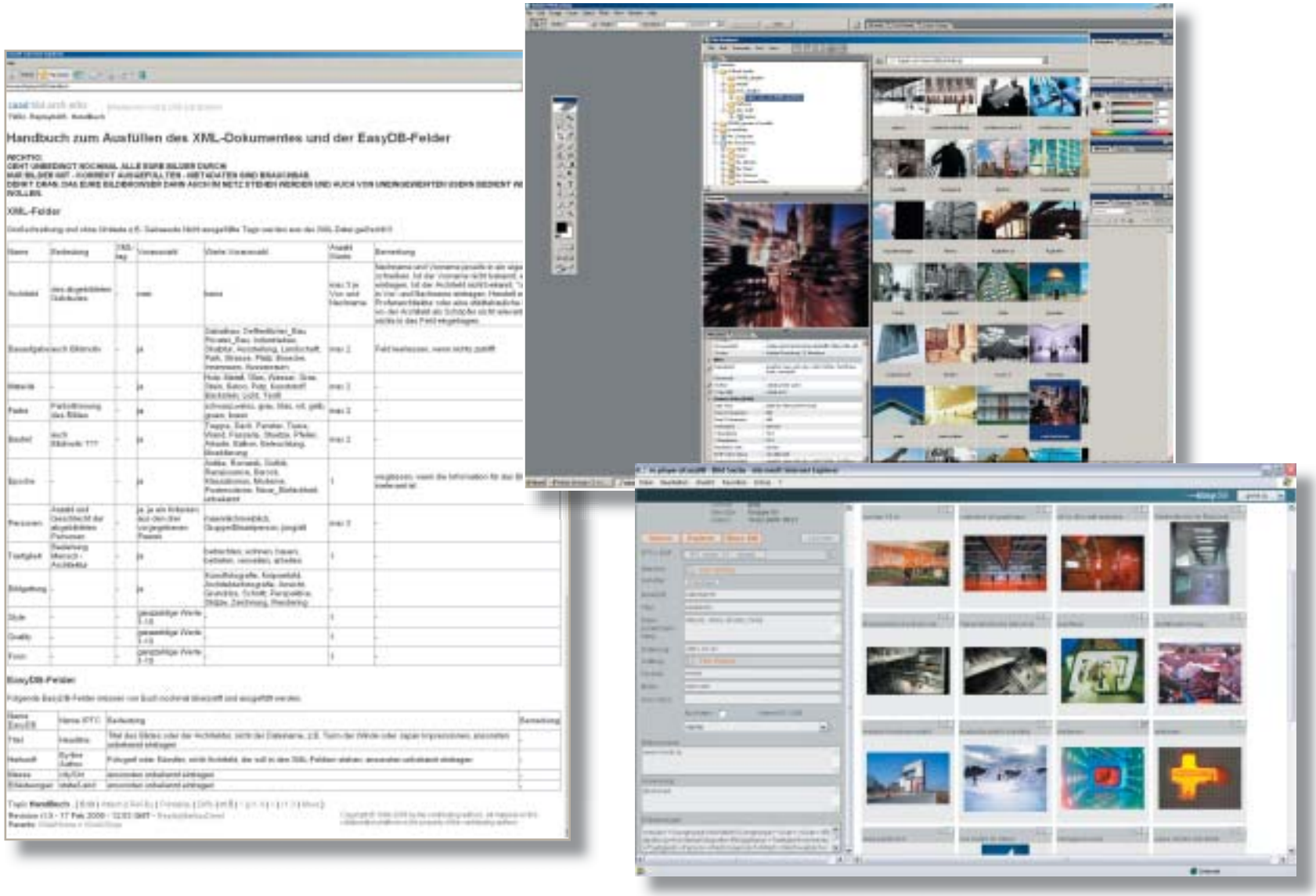
```
                    this._alpha = 100;
                };
                _root.ferdiParent_mc["ferdi_mc"+i].onRollOut = function() {
                    this._alpha = 40;
                };
        }

};


    // bedingungen fuer die skalierung der bildcontainer.
arrangeImages = function (myPosInArray) {
        var lastBreitehöhe, currentImage;
        currentImage = _root.peter_mc;
        lastImage = _root.stephan_mc;
        lastBreitehöhe = _global.imageControl_array[myPosInArray].width/_global.imageControl_array[myPosInArray].height;
        newX = lastImage._x+scale*360*lastBreitehöhe-120;
        //trace("arrangeImages old x: "+currentImage._x+" new:"+newX);
        currentImage._x = newX;
};


    // reagieren des linken bildcontainers auf anklicken eines thumbnails, eines wertesliders.
setBigImage = function (index,aktion) {
        _root.stephan_mc.setMask(_root.mask3_mc);
        textkill = 0;
        _root.stephan_mc.myindex = index;
        _root.stephan_mc._x = 50;
        _root.stephan_mc._y = 170;
        trace("------------------------------------ setBigImage");
        if(aktion != false){
            aktion=true;
            }
        //_root.peter_mc._x = 320;
        //_root.peter_mc._y = 170;
        if(aktion){
        zufall = _root.randRange(0,_root.getNextImage_array.length-1);
        //_root.peter_mc.image.loadMovie(_global.imageControl_array[index].urlMed, 1);
        trace("_global.imageControl_array.length: " +_global.imageControl_array.length);
        trace("_root.getNextImage_array.length: " +_root.getNextImage_array.length);
        temp_array = _root.getNextImage_array.slice(zufall-1,zufall);
```

```
		trace("temp_array.length: " +temp_array.length);
		temp2_array = _global.imageControl_array.concat(temp_array);
		trace("temp2_array.length: " +temp2_array.length-1);
		delete _global.imageControl_array;
		_global.imageControl_array = new Array();
		_global.imageControl_array = temp2_array.slice();
		trace("_global.imageControl_array.length: " +_global.imageControl_array.length);

		_root.stephan_mc.myindex = _global.imageControl_array.length -1;
		zahl = _root.stephan_mc.myindex;
		}
		else{
			zahl = index;
		}

		trace("_global.imageControl_array.length: " +_global.imageControl_array.length-1);
		_root.stephan_mc.image.loadMovie(_global.imageControl_array[zahl].urlMed, 1);
		stephanForm_mc.slide_mc._x = _global.imageControl_array[zahl].myReplay.Form_Wert[0]*10;
		stephanQuality_mc.slide_mc._x = _global.imageControl_array[zahl].myReplay.Qu_Wert[0]*10;
		stephanStyle_mc.slide_mc._x = _global.imageControl_array[zahl].myReplay.Sty_Wert[0]*10;
		if ((_global.imageControl_array[zahl].orig_height/_global.imageControl_array[zahl].orig_width)<1) {
			_root.stephan_mc._yscale = 50*_global.imageControl_array[zahl].orig_width/_global.imageControl_array[zahl].orig_height;
			_root.stephan_mc._xscale = 50*_global.imageControl_array[zahl].orig_width/_global.imageControl_array[zahl].orig_height;
		} else {
			_root.stephan_mc._yscale = 50;
			_root.stephan_mc._xscale = 50;
		}

		metaupdate(index);
};
	// reagieren des rechten bildcontainers auf anklicken eines thumbnails, eines wertesliders.
setBigImage2 = function (index,aktion) {
		trace("------------------------------------ setBigImage2");
		if(aktion != false){
			aktion=true;
			}
		//_root.peter_mc._x = 320;
		//_root.peter_mc._y = 170;
		if(aktion){
		zufall = _root.randRange(0,_root.getNextImage_array.length-1);
		//_root.peter_mc.image.loadMovie(_global.imageControl_array[index].urlMed, 1);
		trace("_global.imageControl_array.length: " +_global.imageControl_array.length);
		trace("_root.getNextImage_array.length: " +_root.getNextImage_array.length);
```

```
		temp_array = _root.getNextImage_array.slice(zufall-1,zufall);
		trace("temp_array.length: " +temp_array.length);
		temp2_array = _global.imageControl_array.concat(temp_array);
		trace("temp2_array.length: " +temp2_array.length-1);
		delete _global.imageControl_array;
		_global.imageControl_array = new Array();
		_global.imageControl_array = temp2_array.slice();
		trace("_global.imageControl_array.length: " +_global.imageControl_array.length);

		_root.peter_mc.myindex = _global.imageControl_array.length -1;
		zahl = _root.peter_mc.myindex;
		}
		else{
			zahl = index;
		}

		trace("_global.imageControl_array.length: " +_global.imageControl_array.length-1);
		_root.peter_mc.image.loadMovie(_global.imageControl_array[zahl].urlMed, 1);
		peterForm_mc.slide_mc._x = _global.imageControl_array[zahl].myReplay.Form_Wert[0]*10;
		peterQuality_mc.slide_mc._x = _global.imageControl_array[zahl].myReplay.Qu_Wert[0]*10;
		peterStyle_mc.slide_mc._x = _global.imageControl_array[zahl].myReplay.Sty_Wert[0]*10;
		if ((_global.imageControl_array[zahl].orig_height/_global.imageControl_array[zahl].orig_width)<1) {
			_root.peter_mc._yscale = 50*_global.imageControl_array[zahl].orig_width/_global.imageControl_array[zahl].orig_height;
			_root.peter_mc._xscale = 50*_global.imageControl_array[zahl].orig_width/_global.imageControl_array[zahl].orig_height;
		} else {
			_root.peter_mc._yscale = 50;
			_root.peter_mc._xscale = 50;
		}
		metaupdate(index);
		_root.stephan_mc._x = 30;
		arrangeImages(index);
};


		// translator: uebersetzt die stichwoerter der easy-DB in die gewuenschte oberflaechengraphik des sqf-bildbrowsers.
translator = new Object();
translator.city = "masse";
translator.source = "bildnachweis";
translator.keyword = "werkzusammenhang";
translator.author = "herkunft";
translator.date = "datierung";
translator.state = "erlaeuterungen";
translator.architect = "Architekt_nach";
```

metaDaten

| ... als metadaten oder metainformationen bezeichnet man allgemein daten, die informationen über andere daten enthalten. bei den beschriebenen daten handelt es sich oft um größere datensammlungen (dokumente) wie buecher, datenbanken oder dateien. eine allgemeingültige unterscheidung zwischen metadaten und normalen daten existiert allerdings nicht. so werden auch angaben von eigenschaften eines objektes (beispielsweise personennamen) als metadaten bezeichnet. waehrend der begriff “metadaten” relativ neu ist, ist sein prinzip unter anderem jahrhundertelange bibliothekarische praxis. ... | *wikipedia*



Handbuch zum Ausfüllen des XML-Dokumentes und der EasyDB-Felder

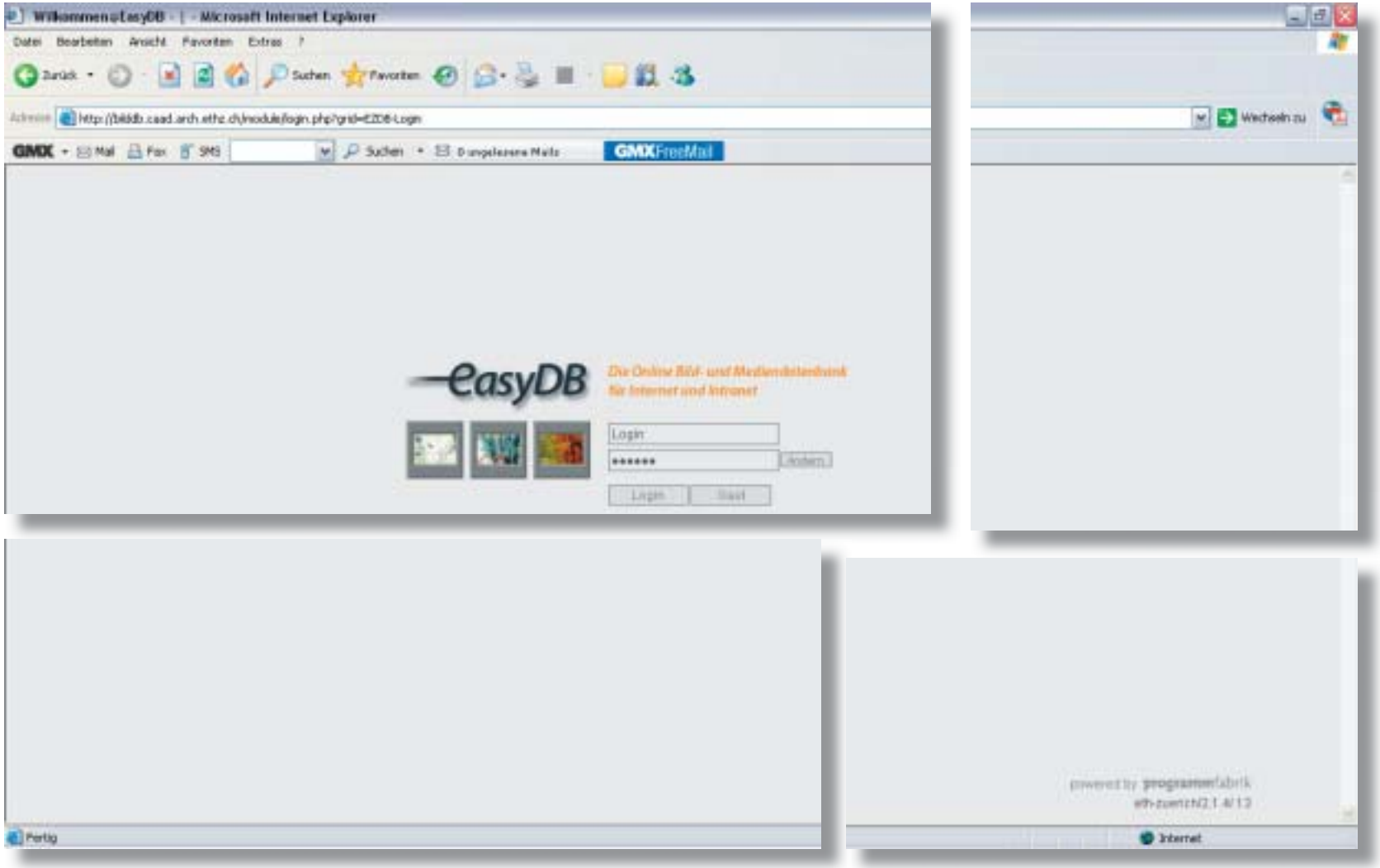
XML-Felder

EasyDB-Felder

easyDB

| ... easyDB ist eine bilddatenbank der neuesten generation für intra- und internet. mit easyDB lassen sich digitale bilddaten einfach verwalten, recherchieren und präsentieren. easyDB basiert auf den open-source-tools Apache, MySql, PHP und kann einfach installiert werden. easyDB kann direkt in einem web-browser genutzt werden. die datenbank orientiert sich für die recherche von bildern an der einfachheit der bekannten suchmaschine “google(tm)” und steht damit einem großen benutzerkreis in unternehmen oder institut, lehranstalten ohne einarbeitungszeit zur verfügung. die vereinnahmung, beschriftung und recherche der bilder erfolgt direkt am arbeitsplatz über den web-browser. dazu wird eine lokal verfügbare bilddatei zum server geladen und dort in die datenbank aufgenommen. die bedienung ist simpel und kann innerhalb kürzester zeit erlernt werden ... |

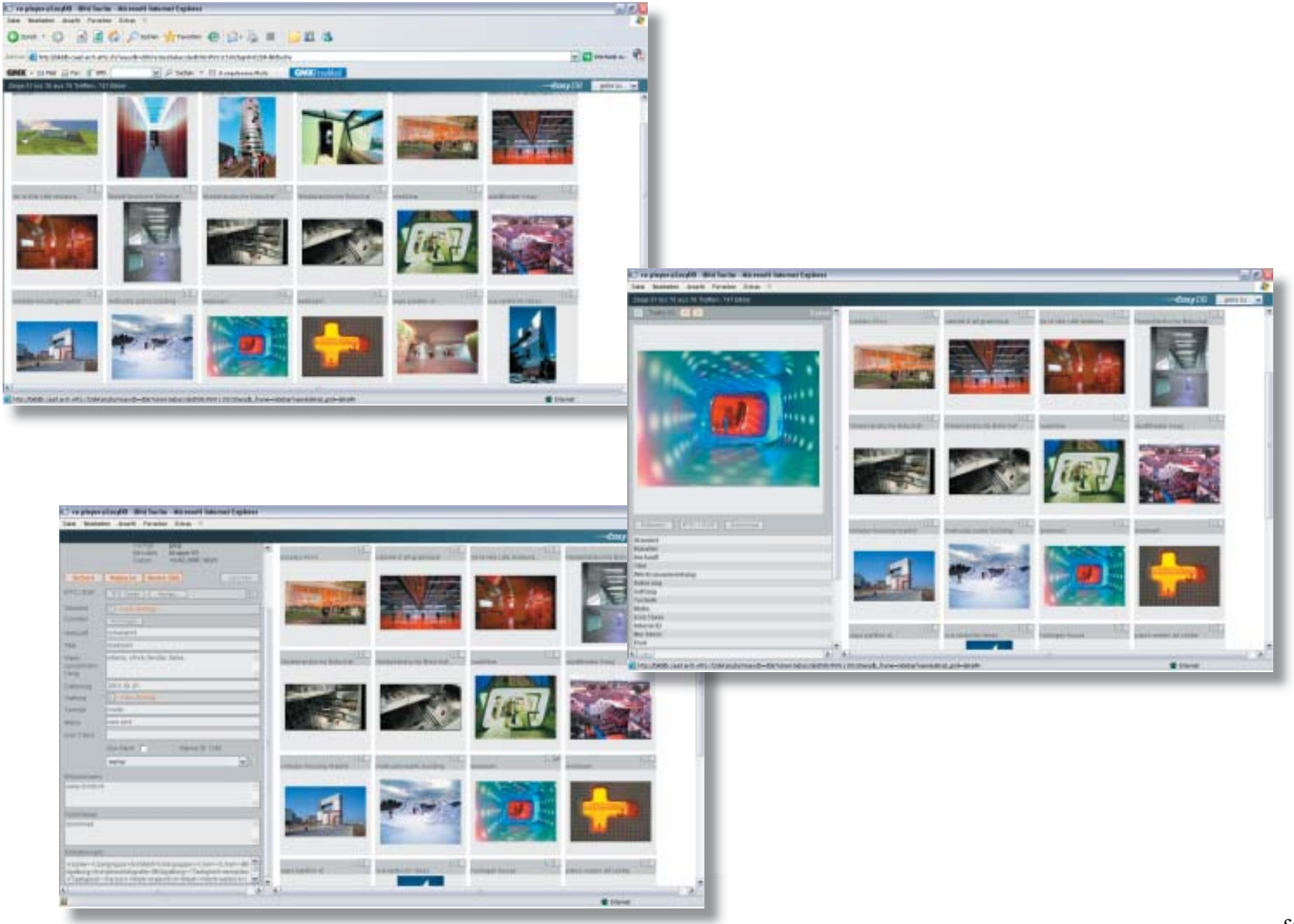
www.easydb.de


Willkommen@EasyDB - [ ] - Microsoft Internet Explorer
Datei Bearbeiten Ansicht Favoriten Extras ?
Zurück
Suchen
Favoriten
GMX
Mail
Fax
SMS
Suchen
GMX FreeMail
EasyDB
Die Online Bild- und Medienbestandsverwaltung
für Internet und Intranet
Login
Fertig
Internet

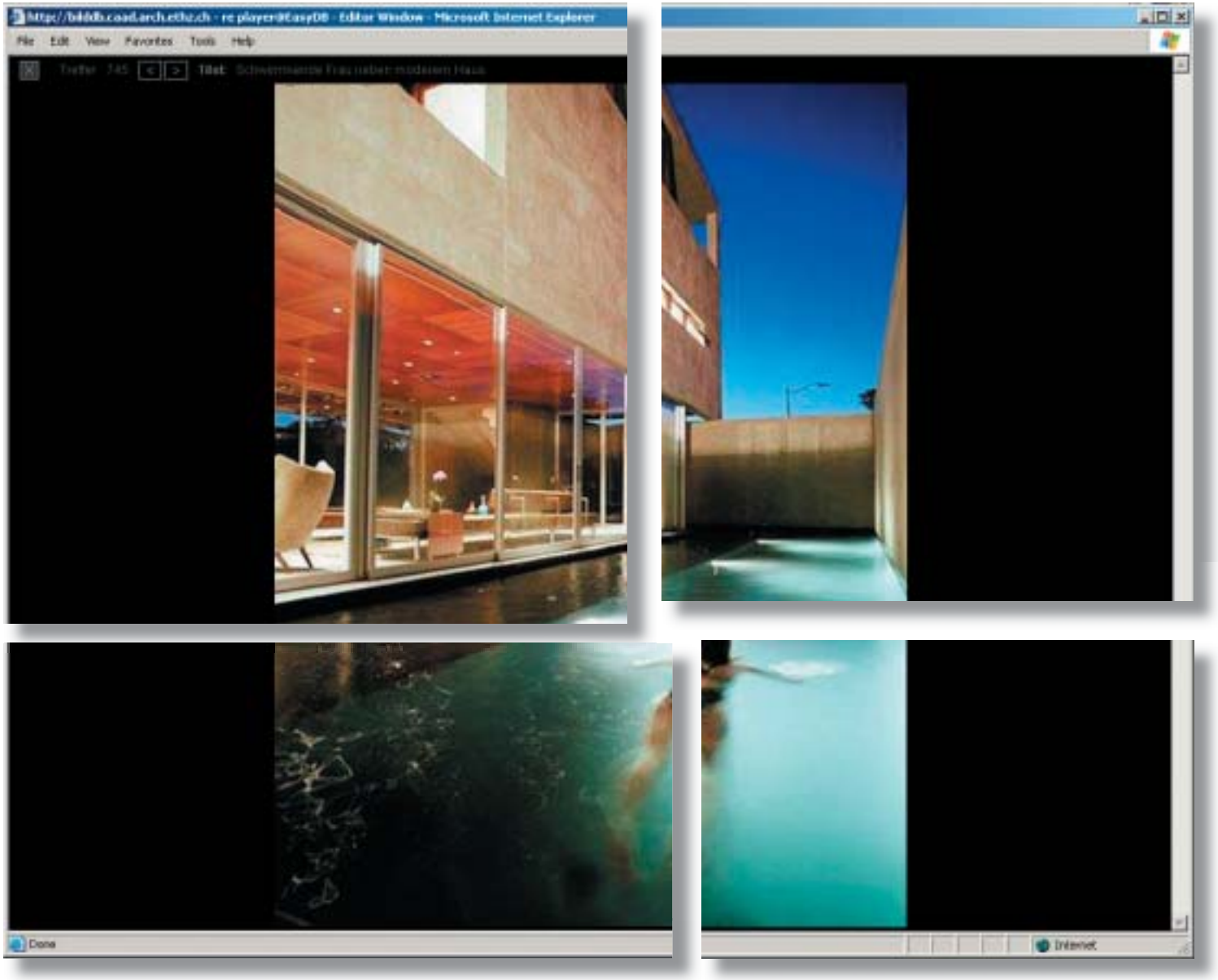
| ... dies ist das standard-suchinterface mit einigen treffern. von hier aus können die bilder editiert, in mappen kopiert, oder im detail angesehen werden ... |

| ... die linke seite des bildschirms (die easyDB-sidebar) wir genutzt, um bilder zu editieren, mappen anzulegen oder im detail auf ein bilddatensatz zu schauen. die rechte seite steht dabei weiterhin fuer die suche zur verfuegung ... |

| ... einen blick auf den editor. auch hier wird die sidebar genutzt. im editor werden alle relevanten informationen zum bild eingegeben. diese kunstgeschichtliche ausprägung der easyDB ist nur ein beispiel. die felder werden an die jeweiligen beduefnisse angepasst ... | *www.easydb.de*



| ... durch doppelklicken kann man sich das bild auf den vollen screen projezieren lassen. zusaetzlich wird die trefferanzahl angezeigt. man kann auf dieser oberflaeche direkt zum naechsten bild springen, ohne die vollscreen-oberflaeche zu verlassen ... |


http://bildb.caad.arch.ethz.ch - re player@EasyDB - Editor Window - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
Internet

xml & flas

sh resources

#include “connectEasyDB_v1.0.as”

```
// ********************************************************************************
// utility actionscript: easyDB communication
// kai rüdenauer : ruedenauer@hbt.arch.ethz.ch
// 22. februar 2005
// filename: connectEasyDB_v1.0.as version 1.0
// ********************************************************************************
// DEFINITION OF VARIABLES
// ********************************************************************************
// _global.imageControl_array
// ******************************************
// Hier werden alle ‘Bilder’ Metadaten abgelegt.
// Jedes Arrayfeld beinhaltet ein loadVars Object.
_global.imageControl_array = new Array();
// DEFINITION OF FUNCTIONS
// ********************************************************************************
// PUBLIC FUNCTION: getInfoEasyDB(myColumn, myTargetArray)
// ********************************************************************************
getInfoEasyDB = function (myColumn, myTargetArray, myTargetFunction) {
	if (typeof (myTargetFunction) != “function”) {
		trace(“Es wurde keine aufzurufende Funktion mitgeschickt. Die Funktion ‘_root.actionOnGetInfoEasyDB()’ wird am Ende der Anfrage aufgerufen!”);
		myTargetFunction = _root.actionOnGetInfoEasyDB;
	}
	// uebersetzen
	if (translator) {
		if (translator[myColumn]) {
			myColumn = translator[myColumn];
		}
	}
	trace(“You called the function: getInfoEasyDB”);
	var getInfo_lv:LoadVars = new LoadVars();
	getInfo_lv.column = myColumn;
	getInfo_lv.mode = “distinct”;
	var getInfo_xml:XML = new XML();
	getInfo_xml.ignoreWhite = true;
	trace(“send request: http://leonbattista.ethz.ch/~spindler/replay-easydb.php?”+getInfo_lv.toString());
	getInfo_lv.sendAndLoad(“http://leonbattista.ethz.ch/~spindler/replay-easydb.php?”+getInfo_lv.toString(), getInfo_xml);
	getInfo_xml.onLoad = function() {
		if (getInfo_xml.firstChild.nodeName == “easydb”) {
			parseInfoXml(getInfo_xml.firstChild, myTargetArray, myTargetFunction);
			_root.error.text = “The easyDB answer is valid”;
			trace(“The easyDB answer is valid”);
		} else {
			_root.error.text = “The easyDB answer is NOT valid”;
			trace(“The easyDB answer is NOT valid”);
		}
	};
};
```

```
// PRIVAT FUNCTION: parseInfoXml(my_xml)
// ****************************************
parseInfoXml = function (my_xml, myTargetArray, myTargetFunction) {
	meineKinder = my_xml.firstChild.firstChild;
	trace("The XML is getting parsed : .."+meineKinder);
	while (meineKinder) {
		myTargetArray.push(String(meineKinder.firstChild));
		meineKinder = meineKinder.nextSibling;
	}
	trace("The target array.lenght is: "+myTargetArray.length);
	// _root.actionOnGetInfoEasyDB();
	trace("Fertig!");
	// ************funktioniert mit ganzem Pfad, z.B. "_root.mySecFunction"
	// [myTargetFunction]();
	// ************funktioniert mit Funktionsnamen wenn die Funktio uaf _root liegt , z.B. "mySecFunction"
	// [myTargetFunction]();
	// ########## Funktioniert auch wird angewendet!!
	trace("The function, which you sended is called..now!");
	myTargetFunction(myTargetArray);
};
// PUBLIC FUNCTION: askEasyDB(myColumn, myArg, myStyle, myTargetArray, myMode, myLimit [,myMovieClip])
// ********************************************************************************
askEasyDB = function (myColumn:String, myArg:String, myStyle:String, myTargetArray:Array, myTargetFunction:Function, myMode:String, myLimit, myMovieClip) {
	// askEasyDB = function (myColumn, myArg, myStyle, myTargetArray, myTargetFunction, myMode, myLimit) {
	if (typeof (myTargetFunction) != "function") {
		trace("Es wurde keine aufzurufende Funktion mitgeschickt. Die Funktion '_root.actionOnAnswer_xmlLoad()' wird am Ende der Anfrage aufgerufen!");
		myTargetFunction = _root.actionOnAnswer_xmlLoad;
	}
	if (myArg.length<1) {
		trace(" Die Anfrage wurde nicht gesendet, da das Anfrageargument keinen Inhalt hatte ..");
		return;
	}
	// uebersetzen
	if (translator) {
		if (translator[myColumn]) {
			myColumn = translator[myColumn];
		}
	}
	if (myTargetArray) {
		trace("Es wurde in der Anfrage ein targetArray übergeben.");
	} else {
		trace("Es wurde in der Anfrage KEIN targetArray übergeben. Das AnfrageErgebnis wird im '_global.imageControl_array' abgelegt!");
		if (_global.imageControl_array) {
			myTargetArray = _global.imageControl_array;
```

```
} else {
                    _global.imageControl_array = new Array();
                    myTargetArray = _global.imageControl_array;
                }
        }
        // initialisieren des LoadVars Objects
        var question_lv:LoadVars = new LoadVars();
        trace("myLimit is: "+myLimit);
        // ***** einfache/normale Anfrage *****
        if (myMode != "complex") {
                if (typeof (myLimit) == "undefined") {
                        myLimit = "";
                }
                question_lv.limit = myLimit;
                question_lv.mode = myMode;
                question_lv.column = myColumn;
                question_lv.search = myArg;
                question_lv.searchstyle = myStyle;
                var answer_xml:XML = new XML();
                answer_xml.ignoreWhite = true;
                question_lv.sendAndLoad("http://leonbattista.ethz.ch/~spindler/replay-easydb.php?"+question_lv.toString(), answer_xml);
                trace("Anfrage: http://leonbattista.ethz.ch/~spindler/replay-easydb.php?"+question_lv.toString());
        }
        // ***** complexe Anfrage *****
        if (myMode == "complex") {
                question_lv.mode = myMode;
                question_lv.where = myArg;
                var answer_xml:XML = new XML();
                answer_xml.ignoreWhite = true;
                question_lv.sendAndLoad("http://leonbattista.ethz.ch/~spindler/replay-easydb.php?"+question_lv.toString(), answer_xml);
                trace("Anfrage: http://leonbattista.ethz.ch/~spindler/replay-easydb.php?"+question_lv.toString());
        }
        answer_xml.onLoad = function() {
                if (answer_xml.firstChild.nodeName == "easydb") {
                        parseMyXml(answer_xml.firstChild, myTargetArray, myTargetFunction, myMovieClip);
                        _root.error.text = "The easyDB answer is valid";
                        trace("The easyDB answer is valid");
                } else {
                        _root.error.text = "The easyDB answer is NOT valid";
                        trace("The easyDB answer is NOT valid");
                }
        };
};
```

```
// PRIVAT FUNCTION: parseMyXml(my_xml, myTargetArray, myTargetFunction[, myMovieClip])
// ******************************************
parseMyXml = function (my_xml, myTargetArray, myTargetFunction, myMovieClip) {
        if (my_xml.nodeName != "easydb") {
                parseMyXml(my_xml.firstChild, myTargetArray);
        }
        _root.extractXML(my_xml, myTargetArray, myTargetFunction, myMovieClip);
};
// PRIVAT FUNCTION: extractXML(my_xml, myTargetArray, myTargetFunction[, myMovieClip])
// ******************************************
extractXML = function (my_xml, myTargetArray, myTargetFunction, myMovieClip) {
        if (my_xml.firstChild.nodeName == "picture") {
                for (var aNode = my_xml.firstChild; aNode != null; aNode=aNode.nextSibling) {
                        storeResult(aNode, myTargetArray);
                }
                // hier können Funktionen aufgerufen werden, welche ausgeführt werden sollen
                // nach der Ankunft einer Antwort der easyDB.
                // **** besser ist es die auszuführenden Aktion oder Funktionen in der Funktion actionOnAnswer_xmlLoad aufzurufen.****
                trace("Die Anfrage ergab "+myTargetArray.length+" Treffer!");
                trace("The function, which you sended or not is called.. now!");
                // parsen des replay tags und einlesen in das Feld myReplay
                // die myTargetFunction(myTargetArray, myMovieClip) Funtion wird in 'parseMyReplay' aufgerufen
                parseMyReplay(myTargetArray, myTargetFunction, myMovieClip);
                //
        }
};
// PRIVAT FUNCTION: storeResult(my_xml, myTargetArray)
// ******************************************
// die einzelnen Werte werden in das LoadVars Object geschrieben, welches
// in einem Feld des myTargetArray sitzt ( wird jeweils hinten angefügt, .. push())
// Wo werden die Daten gespeichert?
//
// ---- myTargetArray[i]
//                                          |
//                                          LoadVars Object
//          ..[i].wertname = wert
storeResult = function (my_xml, myTargetArray) {
        meineKinder = my_xml.firstChild;
        var toPush_lv:LoadVars = new LoadVars();
        meineKinder = my_xml.firstChild;
        while (meineKinder) {
                if (meineKinder.nodeName == "replay") {
                        toPush_lv[meineKinder.nodeName] = String(meineKinder);
                        meineKinder = meineKinder.nextSibling;
```

```
} else {
                                        toPush_lv[meineKinder.nodeName] = String(meineKinder.firstChild);
                                        meineKinder = meineKinder.nextSibling;
                                }
        }
        myTargetArray.push(toPush_lv);
};
// PRIVAT FUNCTION: parseMyReplay(myTarget_array, myTargetFunction, myMovieClip)
// ********************************************************************************
// Die 'String' Information im Replay Feld wird in ein XML umgewandelt,
// die Informationen Transformiert und im myTarget_array[i].myReplay zugänglich gemacht
parseMyReplay = function (myTarget_array:Array, myTargetFunction:Function, myMovieClip) {
        for (z=0; z<myTarget_array.length; z++) {
                myTarget_array[z].myReplay = new LoadVars();
                var my_xml:XML = new XML();
                my_xml.ignoreWhite = true;
                my_xml.parseXML(myTarget_array[z].replay);
                extractReplay(my_xml.firstChild, myTarget_array, z);
        }
        // Aufruf der Funktion, nachdem alle Daten eingelesen sind.
        myTargetFunction(myTarget_array, myMovieClip);
};
// PRIVAT FUNCTION: extractReplay(my_xml, myTarget_array, index)
// ********************************************************************************
extractReplay = function (my_xml, myTarget_array, index) {
        for (var aNode = my_xml.firstChild; aNode != null; aNode=aNode.nextSibling) {
                storeValue(aNode, myTarget_array, index);
        }
};
```

```
// PRIVAT FUNCTION: storeValue(my_xml, myTarget_array, index)
// ********************************************************************************
// die einzelnen Werte werden in das LoadVars Object geschrieben, welches
// in einem Feld des myTargetArray sitzt ( wird jeweils hinten angefügt, .. push())
// Wo werden die Daten gespeichert?
//
// ---- myTargetArray[i]
//                                              |
//                                              LoadVars Object
//                                                      |
//                                                                                      LoadVars Object im Feld 'myReplay'
//                                                                                                                              |
//                                                                                                                      unter jedem Wertnamen ist ein Array[]
```

```
//
  |
//
                              wert
//          zum Beispiel      ..[i].myReplay.epoche[1]
storeValue = function (my_xml, myTarget_array, index) {
            // myTarget_array[i].myReplay = new LoadVars();
            temp_array = myTarget_array[index].myReplay[my_xml.nodeName]=new Array();
            if (my_xml.firstChild.nodeName == "Wert") {
                        extractChilds(my_xml, temp_array);
            } else {
                        if (my_xml.firstChild != null) {
                                    myTarget_array[index].myReplay[my_xml.nodeName].push(String(my_xml.firstChild));
                        }
            }
};
// PRIVAT FUNCTION: extractChilds(my_xml, temp_array)
// ********************************************************************************
extractChilds = function (my_xml, temp_array) {
            for (var aNode = my_xml.firstChild; aNode != null; aNode=aNode.nextSibling) {
                        if (aNode.firstChild != null) {
                                    temp_array.push(String(aNode.firstChild));
                        }
            }
};
// PRIVAT FUNCTION: myReplayToString(my_lv)
// ********************************************************************************
// zum Beispiel: actReplay_txt = prepareSend(beispielArray[i].myReplay);
myReplayToString = function (my_lv) {
            my_text = "<replay>";
            for (name in my_lv) {
                        my_text += "<"+name+">";
                        if (my_lv[name].length == 1) {
                                    my_text += my_lv[name][0];
                        } else {
                                    for (h=0; h<my_lv[name].length; h++) {
                                                my_text += "<Wert>"+my_lv[name][h]+"</Wert>";
                                    }
                        }
                        my_text += "</"+name+">";
            }
            my_text += "</replay>";
            trace("------------------- myReplayToString - - - -..");
            trace(my_text);
            return my_text;
};
// PUBLIC FUNCTION: searchMyReplay(my_array, myColumn, myValue):ARRAY
```

```
// ********************************************************************************
// zum Beispiel: resultSearchReplay_array = searchMyReplay(an_array, “Material”, “Stein”);
searchMyReplay = function (my_array, myColumn, myValue) {
        myResult_array = new Array();
        for (i=0; i<my_array.length; i++) {
                for (k=0; k<my_array[i].myReplay[myColumn].length; k++) {
                        if (String(my_array[i].myReplay[myColumn][k]) == myValue) {
                                myResult_array.push(my_array[i]);
                        }
                }
        }
        return myResult_array;
};
/

/ PUBLIC FUNCTION: sendEasyDB(my_lv, myColumn, myArg)
// ********************************************************************************
//
sendEasyDB = function (my_lv, myColumn) {
        var tempString:String = new String();
        var tempXML = new XML();
        var send_lv:LoadVars = new LoadVars();
        send_lv.mode = “update”;
        send_lv.id = my_lv.id;

        if ((myColumn=”myReplay”) || (myColumn=”replay”) || (myColumn=”erlaeuterungen”)) {
                trace(“my_lv.erlaeuterungen: “ + my_lv.erlaeuterungen);
                trace(“my_lv.erlaeuterungen: “ + my_lv.erlaeuterungen);
                send_lv.erlaeuterungen = my_lv.erlaeuterungen+ _root.myReplayToString(my_lv.myReplay);
                trace(“send_lv.erlaeuterungen: “ + send_lv.erlaeuterungen);
        } else {
                send_lv[myColumn] = String(my_lv[myColumn]);
        }
        trace(“verschickt: “ + “http://leonbattista/~spindler/replay-easydb.php?”+send_lv.toString());
        send_lv.sendAndLoad(“http://leonbattista/~spindler/replay-easydb.php?”+send_lv.toString(), tempXML);


};
```

actionScript resources

Für den replay Workshop wird euch die Kommunikation mit der Bilderdatenbank ‘easyDB’ zur Verfügung gestellt.
Um diese Funktionalität zu nutzen müsst ihr im ersten Frame eurer Animation die entsprechende ActionScript Datei mit dem #include Befehl einbinden. Zum Beispiel:
#include “connectEasyDB_v0.01.as”AlteVersionen

connectEasyDB_v1.0.as:
Anwendungen:
FUNKTION: _askEasyDB(myColumn:String, myArg:String [, myStyle:String, myTargetArray:Array, myTargetFunction:Function, myMode:String, myLimit])
stellt eine einfache Anfrage an die Datenbank(genau genommen an ein php Script, welches die Kommunikation mit der datenbank übernimmt), zum Beispiel: askEasyDB(“Titel”,”freak”);
die zusätzliche Parameter erlauben jedoch eine genauere Abfrage. Neu ist die Angabe eines Arrays in welches die Informationen abgelegt werden können und die Angabe einer Funktion, die ausgeführt werden soll, wenn die Antwort der Datenbank verarbeitet ist.
die Parameter in der eckigen Klammer sind optional. Will man einen der Parameter benutzen, muessen die vorangehenden Parameter ebenfalls gesetzt werden, mindestens mit dem ‘default’ Wert!
myStyle [default: “normal”]:Sring!
“exact” : gibt nur wirklich eindeutige Suchergebnisse zurück, z.B. _root.askEasyDB(id, 45,”exact”); // max. 1 Ergebnis
“numerical_less” : entspricht der < Anfrage, z.B. _root.askEasyDB(id, 23,”numerical_less”); // mehrer Ergebnisse möglich
“numerical_greater” : entspricht der > Anfrage, z.B. _root.askEasyDB(id, 67,”numerical_less”); // mehrer Ergebnisse möglich
“numerical_less_equal” : entspricht der <= Anfrage, z.B. _root.askEasyDB(id, 48,”numerical_less_equal”); // mehrer Ergebnisse möglich
“numerical_greater_equal” : entspricht der >= Anfrage, z.B. _root.askEasyDB(id, 13,”numerical_greater_equal”); // mehrer Ergebnisse möglich
myTargetArray [default: default]
hier könnt ihr ein Array angeben, in welches das Ergebnis eurer Suchanfrage abgelegt werden soll. Das Array muss VOR der Anfrage schon existieren!
soll das Ergebnis in das _global.imageControl_array abgelegt werden muss der Eintrag default sein.
myTargetFunction [default: default]
hier könnt ihr eine FUNKTION angeben, die ausgeführt werden soll anstelle der _root.actionOnAnswer_xmlLoad(); Funktion. Die myTargetFunction muss schon vor der Anfrage existieren!
soll die übliche root.actionOnAnswer_xmlLoad();_ ausgeführt werden muss der Eintrag default sein.
myMode [default: “normal”]:Sring!
“normal” : ergibt kein Unterschied zur bisherigen Suchanfrage
“complex” : erlaubt das zusammenstellen komplexer Anfragen mit SQL Statements MIT VORSICHT ZU GEBRAUCHEN !!
Der Mode ‘complex’ gibt mit dem Parameter ‘where’ Informationen aus der Datenbank. Damit können nur die gesamten Informationen eines Feldes ausgegeben, Groß- und Kleinschreibung ist verbindlich, Worte werde mit ‘ ‘ eigegeben, Zahlenwerte werden direkt eingegeben. (Beim Suchmodus ‘complex’ wird die Option myStyle überschrieben, muß aber als Parameter mitgesendet werden.)
Z.B. die Suchanfrage für Bilder mit der Größe 100-200kb:
WHERE filesize > 100000 AND filesize < 200000.
Im URL muss das mit Apostroph getrennt sein:
mode=complex&where=”filesize > 100000 AND filesize < 200000”.
z.B.http://leonbattista.ethz.ch/~spindler/replay-test.php?mode=complex&where=”Bilder.id<1724 AND Bilder.id>1720”
Oder um aus dem Feld “Werkzusammenhang” dieses Suchergebnis zu erhalten: “strasse, shinjuku, tokio ,japan, leuchtreklamen” müßt ihr in die Anfrage die gesamten Worte eingeben. Das sieht dann so aus:
http://leonbattista/~spindler/replay-easydb.php?mode=complex&where=”Werkzusammenhang=’strasse, shinjuku, tokio ,japan, leuchtreklamen’”
mylimit
“limit”: Anzahl der zurückgegebenen Bilder wird eingeschränkt
z. B: http://leonbattista.ethz.ch/~spindler/replay-test.php?column=masse&search=budapest&limit=3 // liefert 3 Suchergebnisse
Die Zahl gibt die Anzahl der Bilder an. Mit n,m kann ein Bereich angegeben werden: limit=0,10 zeigt die Bilder 0-10 an, limit=11,20 die Bilder 11-20.
FUNKTION: getInfoEasyDB(myColumn, myTargetAray [, myTargetFunction])
Diese Funktion erlaubt Dir, sehr einfach Informationen aus der Datenbank zu erhalten.
der Parameter in der eckigen Klammer ist optional.
“column”: Auswahl der gewünschten Spalte

myTargetFunction [default: default]
hier könnt ihr eine FUNKTION angeben, die ausgeführt werden soll anstelle der _root.actionOnGetInfoEasyDB Funktion. Die myTargetFunction muss schon vor der Anfrage existieren!
soll die übliche root.actionOnGetInfoEasyDB() , bracuht man den Parameter nicht zu setzen. * myTargetArray
hier MUESST ihr ein Array angeben, in welches das Ergebnis eurer Suchanfrage abgelegt werden soll. Das Array muss VOR der Anfrage schon existieren!
habt ihr z.B. folgende Anfrage verschickt: __root.getInfoEasyDB("masse", _root.meineStadt_array) findet ihr in jedem Feld (Schrank--> Schublade) eine 'Stadt'. z.B. trace(_root.meineStadt_array[2]); // Output: Zuerich
Die Anfrage wird nur mit "mode=distinct" gesendet und gibt nur den Wertebereich einer Spalte wieder. Doppelnennungen sind bereits herausgefiltert.
z.B. http://leonbattista.ethz.ch/~spindler/replay-test.php?mode=distinct&column=masse
FUNKTION: sendEasyDB(my_lv, myColumn)
der zu ändernde Wert muss zuerst im Flashobjekt gespeichert werden, zum Beispiel: _global.imageControl_array[3].titel = "mein neuer titel";
schreibt bzw. speichert Informationen zu einem Bild in der Datenbank
alle beiden Parameter müssen angegeben werden.
my_lv
wo befinden sich die Informationen zu dem Bild, zum Beispiel: _global.imageControl_array[3]
myColumn
Welchen Wert wollt ihr ändern? Für Änderungen an dem 'Replay XML' ist nicht die zu ändernde Eigenschaft anzugeben, sondern immer "myReplay" !
Falsch: _root.sendEasyDB(_global.imageControl_array[3],"Architekt_nach");
Richtig: _root.sendEasyDB(_global.imageControl_array[3],"myReplay");
Richtig: _root.sendEasyDB(_global.imageControl_array[45],"Titel");
Beispiele:
_root.sendEasyDB(meineVariable,"Titel","Dieser Titel ist viel besser");
_root.sendEasyDB(_global.imageControl_array[3],"myReplay");
FUNKTION: searchMyReplay(my_array, myColumn, myValue):ARRAY
mit dieser Funktion könnt ihr das 'replay' xml durchsuchen. NICHT das xml in der Datenbank, die Informationen zu einem Bild müssen zuerst angefragt worden sein.
diese Funktion gibt euch ein Array mit den Treffern zurück.
alle beiden Parameter müssen angegeben werden.
my_array
das Array indem die Informationen abgelegt sind (das Ergebnis der askEasyDB Anfrage), zum Beispiel: _global.imageControl_array
myColumn
In welcher Sparte wollt ihr suchen?
myValue
Nach welchem Wert soll gesucht werden!
Beispiel:
meinErgebnis_array = _root.searchMyReplay(_global.imageControl_array,"Architekt_nach","Gropius");
weitere Informationen:
weitergehende Informationen, z.B. zum _global.imageControl_array oder _root.actionOnAnswer_xmlLoad(); findet ihr hier: AlteVersionen:

replayUtilities_v0.05.as:
Anwendungen:
die Datei muss ebenfalls mit #include "replayUtilities_v0.03.as" eingebunden werden.
in Verbindung mit connectEasyDB_v0.0x.as erlaubt es die automatische Übersetzung des 'Mapping easyDB'
FUNKTION: randRange(min,max)
diese Funktion gibt eine zufällige ganzzahlige Zahl in einem bestimmten Bereich zurück.
siehe auch.
FUNKTION: _root.placeImage(target_mc, myObj, mySpaceHeight:Number, mySpaceWidth:Number, clipName:String, imgUrl:String, posX:Number, posY:Number, center:Boolean, myFunction)
Parameter
target_mc: In welchen Clip soll das Bild geladen werden? Es muss ein MovieClip angegeben werden! z.B. _root.clip_mc

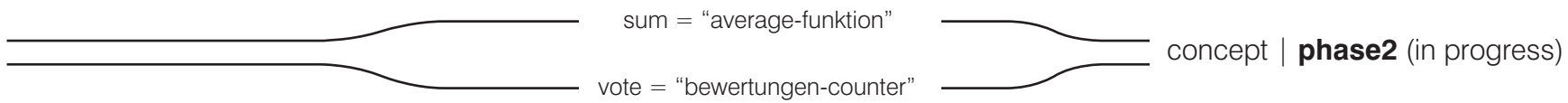
myObj: Wo findet man die Informationen zu dem Bild? z.B. _global.imageControl_array[0]
mySpaceHeight: In welche Höhe soll das Bild eingepasst werden? z.B. 80
mySpaceWidth: In welche Breite soll das Bild eingepasst werden? z.B. 80
clipName: Wie soll der Clip heissen, in welchen das Bild geladen werden soll? z.B. “hans”
--> Information: Das Bild wird nicht in den Clip “hans” geladen. Im Clip “hans” wird ein leerer Clip ‘img_mc’ gemacht, in dem dann das Bild geladen wird.
imgUrl: welche Bildgrösse soll geladen werden? z.B. “urlThumb”
posX: x position im target_mc
posY: y position im target_mc
center: Soll das Bild zentriert werden? z.B. true (HINWEIS: Es muss ein BOOLEAN übergeben werden, kein String!)
myFunction: hier kann man optional eine Funktion übergeben, welche zum Beispiel die Bilder sortiern soll! z.B. “_root.setOrder”
FUNKTION: realSizeOfImage(myObj, imgUrl):LoadVars
diese Funktion gibt euch ein LoadVars Objekt zurück, mit der Bildgrösse des angefragten Bildes in Pixels
über die Eigenschaften toLoadHeight und toLoadWidth kann man die Werte abfragen.
Parameter
myObj: z.Bsp. _global.imageControl_array[3]
imgUrl: z.Bsp. “urlThumb”
Beispiel:
my_lv = _realSizeOfImage(_global.imageControl_array[3], “urlBig”);
trace(_my_lv.toLoadHeight); // Output: 234
trace(_my_lv.toLoadWidth); // Output: 600
FUNKTION: scaleFactor(mySpaceWidth, mySpaceHeight, toLoadWidth, toLoadHeight):NUMBER
diese Funktion gibt euch den passenden Skalierfaktor zurück
Parameter
mySpaceWidth und mySpaceHeight: die Grösse in welche das Bild eingepasst werden soll..
toLoadWidth und toLoadHeight: wie gross idt das zu ladende Bild in Pixel.
Beispiel:
myScale = _root.scaleFactor(456, 234, 678, 267);
myScale = _root.scaleFactor(myWidth, myHeight, _realSizeOfImage(myObj, imgUrl).toLoadWidth, _realSizeOfImage(myObj, imgUrl).toLoadHeight);

Beispiel:
example_1.0_0.05.fla

Replay Downloads

```
function randRange(min,max)
      // The following example returns a random number between two specified integers.
      // Quelle: macromedia flash help: math.random();
      function randRange(min:Number, max:Number):Number {
          var randomNum:Number = Math.round(Math.random()*(max-min))+min;
          return randomNum;
      }
      Description Feld: XML-Struktur
Ausfüllen: siehe Handbuch
<replay>
  <Bauaufgabe>
    <Wert></Wert>
    <Wert></Wert>
  </Bauaufgabe>
```

```
    <Architekt_vorn>
     <Wert></Wert>
     <Wert></Wert>
     <Wert></Wert>
   </Architekt_vorn>
   <Architekt_nach>
     <Wert></Wert>
     <Wert></Wert>
     <Wert></Wert>
   </Architekt_nach>
   <Material>
     <Wert></Wert>
     <Wert></Wert>
   </Material>
   <Bauteil>
     <Wert></Wert>
     <Wert></Wert>
   </Bauteil>
   <Epoche></Epoche>
   <Sty_Wert>5</Sty_Wert>
   <Sty_Sum></Sty_Sum>
   <Sty_Vote></Sty_Vote>
   <Qu_Wert>5</Qu_Wert>
   <Qu_Sum></Qu_Sum>
   <Qu_Vote></Qu_Vote>
   <Form_Wert>5</Form_Wert>
   <Form_Sum></Form_Sum>
   <Form_Vote></Form_Vote>
   <Farbe>
     <Wert></Wert>
     <Wert></Wert>
     <Wert></Wert>
   </Farbe>
   <Person>
     <Wert></Wert>
     <Wert></Wert>
     <Wert></Wert>
   </Person>
   <Taetigkeit></Taetigkeit>
   <Bildgattung></Bildgattung>
   <User>
     <Wert></Wert>
     <Wert></Wert>
     <Wert></Wert>
     <Wert></Wert>
     <Wert></Wert>
   </User>
   <Usergruppe>Architekt</Usergruppe>
</replay>
```



appendix

literatur

zu_____bildgebrauch in der architektur

• Architecture and its Image. Four Centuries of Architectural Representation, Montreal 1989.
• Müller-Wulckow, Walter: Architektur 1900-1929in Deutschland. Reprint und Materialien zur Entstehung, Hrsg. Hans-Curt Köster (Band 1) Kontexte. Walter Müller-Wulckow udn die deutsche Architektur von 1900-1930, Hrsg. Gerd Kuhn, Königstein im Taunus, 1999.
• Pare, Richard: Photography and Architecture 1839-1939, Montreal 1982.
• Sachsse, Rolf : Bild und Bau. Zur Nutzung technischer Medien beim Entwerfen von Architektur, 1997.

zu_____das digitale bild

• Pfenninger, Kathryn: Bildarchiv digital / Kathryn Pfenninger. [Hrsg. von der Arbeitsgruppe “Fotografie im Museum” des Museumsverbands Baden-Württemberg ...]. - Esslingen : Museumsverband Baden-Württemberg, 2001.
• Suchbilder. Visuelle Kultur zwischen Algorithmus und Archiven, Wolfgang Ernst, Stefan Heidenreich, Ute Holl (Hg.), Berlin, 2003.
• Warnke, Martin: Digitales Sammeln, in: Archivierungsprozesse. Die Kommunikation der Aufbewahrung, Hedwig Pompe, Leander Scholz (Hg.), 2002.

zu_____art & databases

• Deep Storage. Arsenale der Erinnerung. Sammeln, Speichern, Archivieren in der Kunst, hrsg. von Ingrid Schaffner, München, 1997/1998.
• Information is alive: Art and Theory on Archiving and Retrieving Data, Joke Brouwer, Arjen Mulder (Hrsg.) Nai Publishers, Rotterdam, 2003.
• Interarchive. Archivarische Praktiken und Handlungsräume im zeitgenössischen Kunstfeld, Köln 2002.
• Making Art of Databases, Joke Brouwer, Arjen Mulder, Susan Charlton, Nai Puplishers, Rotterdam, 2003.
• Manovich, Lev: The language of new media, Cambridge, Mass. : MIT Press, 2001.
• Wilson, Stephen: Information Arts. Intersection of Art, Science, and Technology, The MIT Press, 2002.

zu_____weiteren themen

• Crary, Jonathan: Techniken des Betrachtens. Sehen und Moderne im 19. Jh., Dresden, 1996.

links

zu_____relevanten projekten

Schweden-Puzzel von Sandrine Haeberli und Matthias Lehner
Bildscrabble von Matthias Lehner
Zoom von Tom Pawlofsky

zu_____bildsuche im internet

http://pro.corbis.com/ Corbis - stock photography
http://creative.gettyimages.com Getty Images - stock photography
http://images.google.de/Bildsuche bei Goooogle
http://images.search.yahoo.com/ Bildsuche bei Yahoo
http://suche.web.de/search/pichp/ Bildsuche bei Web.de

zu_____kommerziellen bilddatenbanken

http://www.canto.de Cumulus
http://www.fotoware.com FotoStation
http://www.thumbsplus.de ThumbsPlus
http://www.application-systems.de iView Media Pro
http://www.photools.com/ iMatch

zu_____digitalen bildarchiven

http://www.foto.unibas.ch/~rundbrief/sh7_links.htm sehr gute Linkliste zum Thema zusammengestellt von Kathryn Pfenninger
http://www.bildindex.de Bildindex der Kunst und Architektur Foto Marburg
http://www.prometheus-bildarchiv.de/ Das verteilte digitale Bildarchiv für Forschung & Lehre e.V.

zu_____art & database

http://www.agency-computer.com Eigentumsgebrauch-Datenbank
http://www.iniva.org institute of international visual arts, london
darin besonders: x-space http://www.iniva.org/xspace/index#
http://www.mip.at museum in progress

george legrady

http://www.georgelegrady.com/ und http://www.mat.ucsb.edu/~g.legrady/ Homepage
http://www.medienkultur-stuttgart.de/thema02/2archiv/news5/mks_5_legrady.htm Interview
http://www2.kah-bonn.de/1/22/2.htm Tracing (Internet Version)
http://www.pocketsfullofmemories.com/ Pockets Full of Memories (Internet Version)

lev manovich

http://www.manovich.net/ Homepage

weitere kuenstlerische projekte

http://www.communimage.ch/communimage von calc & Johannes Gees
Exactitudes Archiv zu Identitäten von Ari Versluis und Elli Uyttenbroek

institutionen und datenbanken

http://www.zkm.de/ Zentrum für Kunst und Medientechnologie Karlsruhe
http://www.v2.nl/index.php V2 Media Center Rotterdam
http://www.medienkunstnetz.de/Medienkunstnetz Datenbank zur Medienkunst
http://virtualart.hu-berlin.de/start.do Internationale Datenbank für virtuelle Kunst

20/10/04	CAAD Diplomwahlfächer	gemeinsame Vorstellung und Einführung, anschliessend Anmeldung	Übungen
21/10/04	replay_01 **Bildgebrauch in der Architektur**	Auftakt Einführung in den twiki Bildgebrauch in der Architektur_1: Katharina Bosch	Digitaler Kontaktbogen
28/10/04	replay_02 **Bildgebrauch in der Architektur**	Bildgebrauch in der Architektur_2: Katharina Bosch Einführung: Das digitale Bild_1: Torsten Spindler	Sortiermaschinen
04/11/04	replay_03 **Digitales Sammeln**	Kommerzielle Bilddatenbanken Gast: Philipp Schaerer: Wissensmanagement bei Herzog & de Meuron	art & database Flash WarmUp Bildkompression
11/11/04	replay_04 **art & database**	Künstlerische Datenbankprojekte Gäste: Bildsynchron Hansuli Matter und Nicole Schneider stellen das Forschungsprojekt der HGK Zürich / scenographical design vor. Möglichkeiten der Bildannotation Susanne Schumacher	ReplayFlash Uebung 1
18/11/04	replay_05 **Flash**	Das digitale Bild_2: Torsten Spindler ContentBasedImageRetrieval: Katharina Bosch Flash_1: Kai Rüdenauer	-
25/11/04	kein replay	Seminarwoche	-
02/12/04	replay_06 **Flash**	Flash_2: Kai Rüdenauer	ReplayFlash Uebung 2
09/12/04	replay_07 **Flash**	Konzept Bildbrowser IdeenSammlung Flash_3: Kai Rüdenauer	Konzept Bildbrowser
16/12/04	replay_08 **Flash + Bildbrowser**	Metadaten: Susanne Schumacher erste Konzepte für den Bildbrowser Flash_4: Kai Rüdenauer	Bildersammlung Konzept Bildbrowser
24/12/04 - 03/01/04	kein replay	Weihnachtsferien	-
13/01/05	replay_09 **Bildbrowser**	Flash_5 Zwischenpräsentation Konzepte Bildbrowser mit Gästen: Sabine Wolf, Steffen Lemmerzahl und Philipp Schaerer	-
14/02 - 24/02/05	replay_workshop	Entwicklung und Umsetzung Bildbrowser	-
25/02/04	replay Präsentation	Bildbrowser	-
???	Abgabe	Dokumentation Bildbrowser	-