ECSP – Embedded Client Side Paradata*

Stephan Schlosser (University of Göttingen)
Jan Karem Höhne (University of Mannheim)

ECSP introduction

Embedded Client Side Paradata (ECSP) is a tool that is licensed under the Creative Commons Attribution 4.0 International License (see http://creativecommons.org/licenses/by/4.0/).[1] It is based on different program languages, such as JavaScript and HTML. In general, ECSP can be implemented in web-based survey software solutions that provide access to the source code.[2] It enables researchers to passively collect different kinds of client-side paradata, such as response times and scrolling events – irrespective of the Internet browser and operating system used – to investigate respondents' completion behavior with respect to web surveys. Paradata are collected at the page-level and are stored together with the actual survey data (i.e., respondents' answers) in the same dataset.

This contribution represents an updated version of the ECSP tool published by Schlosser (2016). It is an expansion of the 2016 version that introduces six new paradata types. We do not discuss the contents of the previous version from 2016. For more detailed information on the functionality, implementation (for non-optimized survey designs), and usage of ECSP, we refer interested readers to Schlosser (2016). This also applies to the descriptions of program codes published in the previous version, which includes the following paradata types: 1) browser window size, 2) keystrokes (PCs and laptops only), 3) mouse clicks and finger taps, 4) mouse movements, 5) response and transmission times, 6) vertical scrolling, 7) basic SurveyFocus (Höhne & Schlosser, 2017; Höhne, Schlosser, & Krebs, 2017), and 8) zooming.

In this contribution, we introduce and outline the program codes of the following six new paradata types: 1) connection type, 2) device orientation and orientation change, 3) screen resolution and pixel ratio, 4) horizontal scrolling, 5) mobile SurveyFocus (Schlosser & Höhne, 2017), and 6) client-side user agent string. In contrast to the previous version, all ECSP paradata

*This manuscript contains the ECSP codes from 2016 and the new codes from 2018 (see Appendix A and B). For the sake of convenience, we recommend that ECSP users only refer to the new version (Schlosser & Höhne, 2018).

Corresponding author
Stephan Schlosser, Faculty of Social Sciences, Center of Methods in Social Sciences, University of Göttingen, Goßlerstraße 19, Room 1.103, Göttingen 37073, Germany.
Email: sschlos1@uni-goettingen.de

– in this updated version – can be collected for non-optimized and optimized survey designs (i.e., the layout is adapted to the screen size of the device).

The ECSP paradata codes can be customized to suit individual needs. In other words, only paradata types that are deemed necessary can be collected. ECSP users can drop the paradata codes that they do not need without affecting the collection of the remaining paradata types. However, the collection of response and start times is always necessary for collecting time stamps for the other paradata types.

ECSP extension

Connection type

ECSP can register the connection type of devices by distinguishing the following types of connection: WiFi, cellular, undefined, and unknown.[3] However, this registration is only possible for the following Internet browsers: Android Webview (as of version 50), Google Chrome (as of version 61), Google Chrome for Android (as of version 38), Mozilla Firefox Mobile (as of version 12), Mozilla Firefox OS (as of version 1.4), and Opera Mobile (as of version 37).

Device orientation and orientation change

ECSP can register the device orientation and device orientation change (e.g., from portrait to landscape), including time stamps. Information about the orientation and its change can be collected only for devices containing an orientation sensor. ECSP records the following values to indicate the orientation of the device: 0 (portrait), 180 (portrait; upside down)[4], 90 (landscape; left aligned), and –90 (landscape; right aligned). If changes in the Internet browser occur (e.g., respondents leave the web survey page for a certain time; see SurveyFocus), ECSP collects the device orientation again.

Screen resolution and pixel ratio

ECSP can register the screen resolution and pixel ratio of devices. Technically, it records the logical screen resolution (LSR_{xy}) in pixels (e.g., 360 × 640). In addition, ECSP records the pixel ratio (PR) – values are greater than or equal to 1 – to determine the physical screen resolution (PSR). For this purpose, users can simply multiply the LSR_{xy} by the PR. Figure 1 is an example of how to calculate PSR.

$$PSR = LSR_x \cdot PR \times LSR_y \cdot PR$$
$$PSR = 360 \cdot 3 \times 640 \cdot 3$$
$$PSR = 1080 \times 1920$$

Figure 1. PSR calculation

Horizontal scrolling[5]

ECSP can register horizontal scrolling events (e.g., from left to right), including time stamps. It collects information about the pixel shift of the left edge of the browser window. For example, if a respondent scrolls 100 pixels to the right, ECSP records this value. However, for example, if a respondent scrolls the same 100 pixels back to the left, ECSP records 0 (the value of the left browser edge). If the codes for horizontal and vertical scrolling are used together, ECSP continuously collects information for both scrolling types. For example, if a respondent engages in vertical scrolling, ECSP also records the horizontal scrolling position, even if it does not change.

Mobile SurveyFocus

SurveyFocus (SF) was originally developed by Höhne and Schlosser (2017) and Höhne, Schlosser, and Krebs (2017). ECSP can register the in-/activity of the web survey page (i.e., whether the browser window that hosts the web survey was the active or processed one; Callegaro, 2013, p. 269), including time stamps. ECSP records the following values to indicate the in-/activity of the web survey page: 1 (active; SF:ON) and 0 (inactive; SF:OFF). In multi-device surveys (e.g., PCs and smartphones), the ECSP codes of the basic and mobile SurveyFocus (SF) can be used together.

User agent string

ECSP can register the user agent string (UAS) of the devices used. Basically, UASs display information about the device (e.g., manufacturer and model), Internet browser (e.g., provider and version), and operating system (e.g., system and version). Figure 2 is a UAS recorded by ECSP.

```
Mozilla/5.0 (Linux; Android 6.0.1; Nexus 5 Build/M4B30Z) AppleWebKit/537.36
KHTML, like Gecko) Chrome/65.0.3325.109 Mobile Safari/537.36
```

Figure 2. Example of a user agent string
Note. This user agent string belongs to the device of one of the authors of this contribution.

Since ECSP records UASs at the page-level it is possible to register device changes during web survey completion (e.g., switching from a PC to a smartphone).

ECSP limitations

The current paradata transfer occurs synchronically between the clients' Internet browsers and the web survey host (i.e., the server). However, an asynchronous communication via AJAX (Asynchronous JavaScript and XML) would be highly desirable because this would decrease transmission times between clients and the host. In addition, this would enable researchers to collect information about respondents' completion behavior if they do not submit a survey page (e.g., by clicking the "Next" button) because they break-off from the web survey.

A further limitation is that ECSP cannot collect any paradata if respondents have deactivated JavaScript. However, the proportion of respondents that have deactivated JavaScript is comparatively small. Höhne and Schlosser (2017) and Höhne, Schlosser, and Krebs (2017) have shown that this situation applies to less than 1% of all respondents, irrespective of the device type used.

ECSP and ethical considerations

The use of program languages, such as JavaScript, enables researchers to passively collect many paradata without respondents' knowledge and consent, which also applies to the collection of client-side paradata by means of ECSP. Researchers using such data face serious ethical considerations. Thus, although we encourage researchers to make use of paradata to improve and enhance survey research methods, we clearly state that these data should not be used to surveil respondents or to frivolously adapt final responses given by respondents (see Heerwegh, 2002). We are convinced that these kinds of data should not be collected without respondents' consent, even if willingness to participate decreases (see Couper & Singer, 2013). Furthermore, we highly recommend checking (specific) legal prerequisites to protect the online privacy of respondents.

ECSP disclaimer

Although the authors tested the application of ECSP in several (pretest) studies, they wish to state clearly here that the use of all program codes is completely the user's own responsibility. There is no warranty of any kind that the codes work properly, and users are encouraged to test their functionality before utilization. The authors and/or their affiliations cannot be held

responsible for any possible malfunctions and/or damages, even if ECSP is the responsible source.

ECSP implementation for optimized survey designs[6]

We now describe a seven-step procedure to implement ECSP in the survey software solution Unipark (Questback). This implementation procedure includes the following steps: 1) implementing JavaScript code (see Appendix A), 2) generating an invisible user-defined question, 3) generating paradata variables, 4) implementing HTML code (see Appendix B), 5) adapting HTML code, 6) connecting survey page and JavaScript, and 7) functionality test. All steps should be conducted carefully and in the given order. It is recommended that the ECSP tool will be implemented after the actual questionnaire programing.

Steps 1 to 5 must be conducted only for the first questionnaire page because the invisible user-defined question can be simply copied for all the other pages. However, step 6 must be conducted manually for each questionnaire page.

Step 1: Implementing JavaScript code

Select *Page structure of standard questionnaire pages* by accessing the sub menu *Pro editor* (*Layout → Pro editor*).

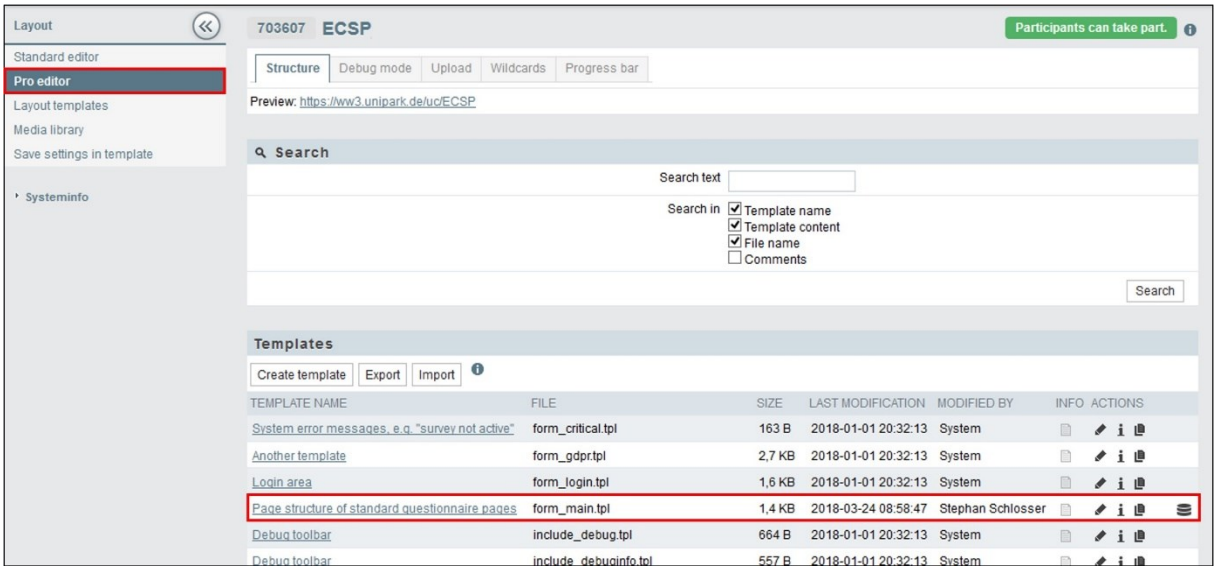

Figure 3. Pro editor and page structure of standard questionnaire pages

Implement the ECSP JavaScript code (see Appendix A) in the source code and click the *Save* button. Do not adapt the main code.

Figure 4. Access point to the JavaScript source code

Step 2: Generating an invisible user-defined question

Access the sub menu *Questionnaire editor* and generate a further question (type: *911 user-defined*) for the first questionnaire page.

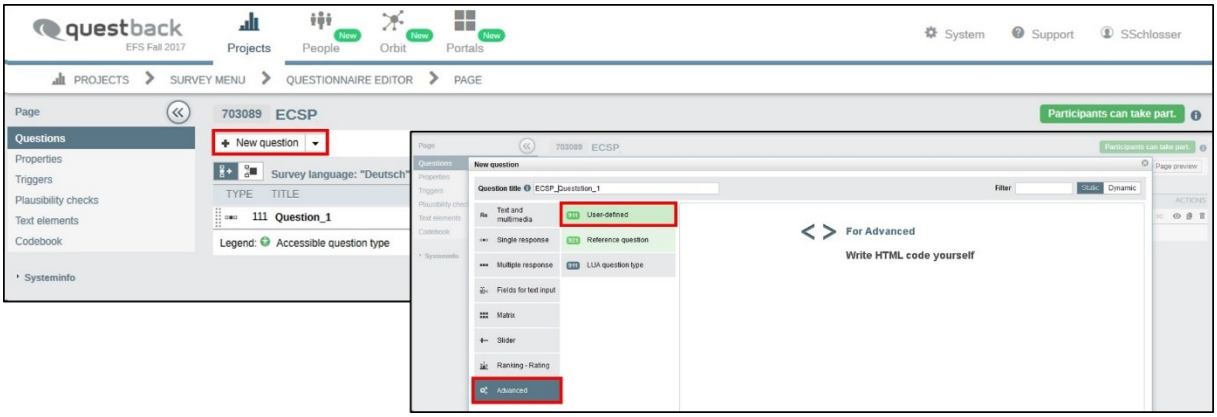

Figure 5. Generating an invisible user-defined question

Step 3: Generating paradata variables

For each paradata type, several variables must be generated. For example, to collect horizontal scrolling events, the following two variables must be generated:

1) SCROLLING (HORIZONTAL) X
2) SCROLLING (HORIZONTAL) TIME

The variable type *Text (max. 65535 characters)* must be used for all paradata types collected by ECSP, except for response and start times. In these cases, the variable type *Integer (approx. -2 bn to approx. 2 bn)* must be used.

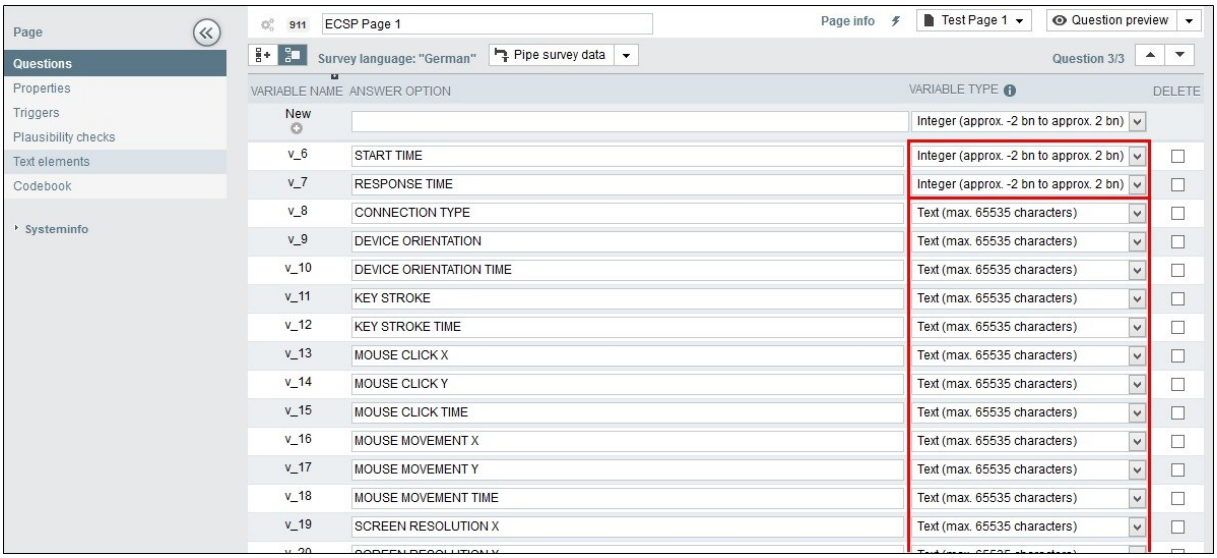

Figure 6. Paradata variables

Step 4: Implementing HTML code

Go to the *Edit HTML* section at the bottom of the *911 user-defined* question page. Implement the ECSP HTML code (see Appendix B).

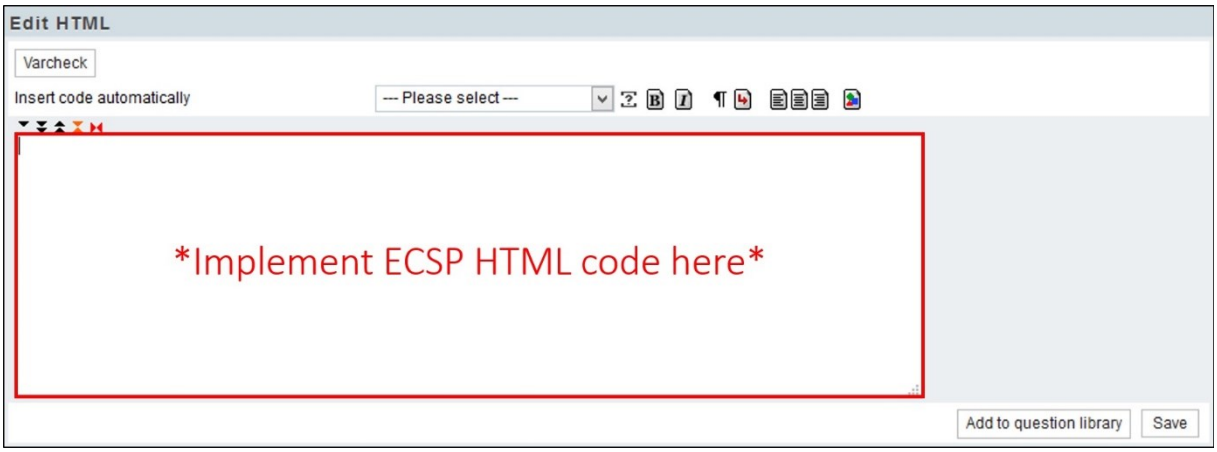

Figure 7. HTML section

Step 5: Adapting HTML code

Adapt the correct variable names given by Unipark (Questback) in the ECSP HTML code. This adaption must be accomplished for all paradata variables. To check the correctness of the variable names, click the *Varcheck* button. Click the *Save* button at the end.

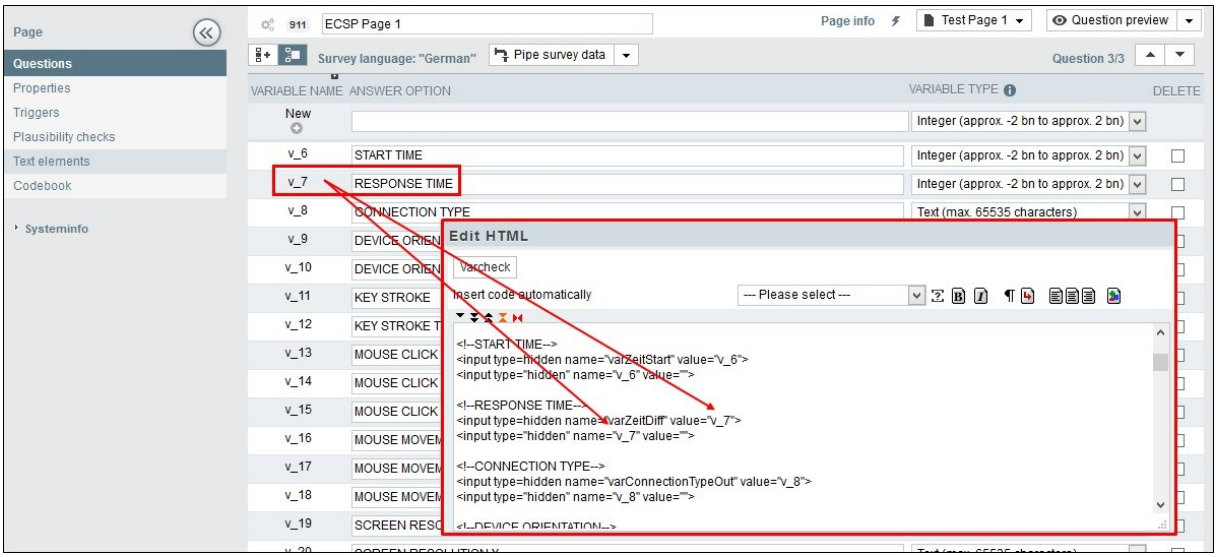

Figure 8. Editing ECSP HTML code

Step 6: Connecting survey page and JavaScript

Access the sub menu *Properties* on the *911 user-defined* question page. Go to the *Additional code* section and enter *send();* in the input field and click the *Save* button.

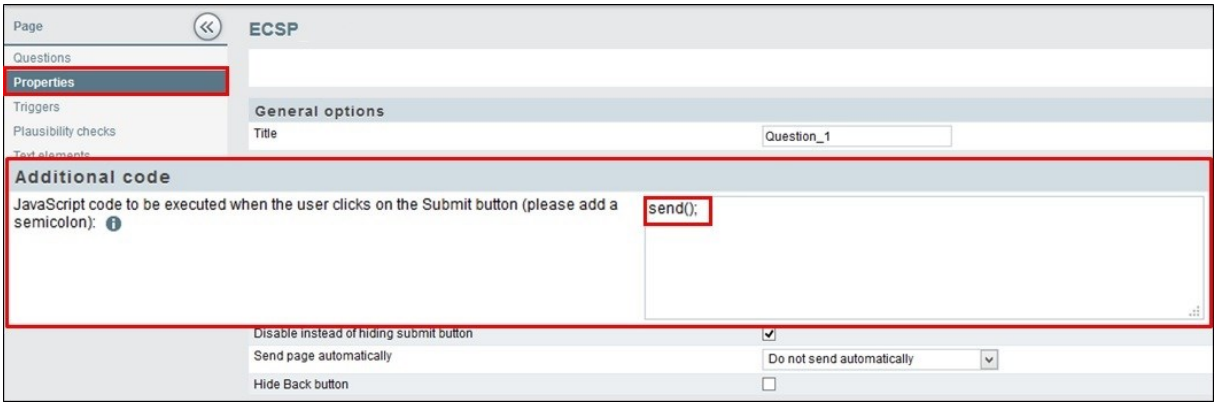

Figure 9. Additional code section

This step must be repeated manually for each *911 user-defined* question page.

Step 7: Functionality test

Conducting functionality tests is highly recommended to make sure that the paradata collection by ECSP is working properly. For example, complete the questionnaire with your smartphone and trigger different paradata events (e.g., changing the device orientation) and control the results.

Endnotes

[1]  Embedded Client Side Paradata (2018) by Stephan Schlosser (University of Göttingen) and Jan Karem Höhne (University of Mannheim) is licensed under the Creative Commons Attribution 4.0 International License. To view a copy of this license, please visit http://creativecommons.org/licenses/by/4.0/.

[2] ECSP was conceptualized for implementation in Unipark (Questback), which is the main survey software solution that the authors use. Therefore, the ECSP implementation procedure is exclusively for Unipark (Questback). Users should keep in mind that they might need to make some adaptions if they wish to use the ECSP program codes for other survey software solutions.

[3] The ECSP version 2018 uses the JavaScript command "connection.type" (see Appendix A). For a more detailed discussion about connection types, we refer interested readers to Couper and Peterson (2017).

[4] This is only possible for a few devices.

[5] In principle, horizontal scrolling is not required in optimized survey designs, although it can be necessary in non-optimized survey designs.

[6] See Schlosser (2016) for ECSP implementation in non-optimized survey designs. All descriptions are based on Unipark (Questback) version 10.9.

References

Callegaro, M. (2013). Paradata in web surveys. In F. Kreuter (Ed.), *Improving Surveys with Paradata. Analytic Uses of Process Information* (pp. 261–280). Hoboken, NJ: John Wiley & Sons.

Couper, M.P., & Peterson, G.J. (2017). Why do web surveys take longer on smartphones? *Social Science Computer Review, 35*, 357–377.

Couper, M.P., & Singer E. (2013). Informed consent for web paradata use. *Survey Research Methods, 7*, 57–67.

Heerwegh, D. (2002). Describing response behavior in web surveys using client side paradata. *Paper presented at the International Workshop on Web Surveys*, Mannheim: Germany.

Höhne, J.K., & Schlosser, S. (2017). Investigating the adequacy of response time outlier definitions in computer-based web surveys using paradata SurveyFocus. *Social Science Computer Review*. DOI: 10.1177/0894439317710450.

Höhne, J.K., Schlosser, S., & Krebs, D. (2017). Investigating cognitive effort and response quality of question formats in web surveys using paradata. *Field Methods, 29*, 365–382.

Schlosser, S. (2016). Embedded Client Side Paradata (ECSP). *Zenodo*. DOI: 10.5281/zenodo.55169.

Schlosser, S. & Höhne, J.K. (2017). Does the continuity of web-survey processing matter? *Paper presented at the Conference of the European Survey Research Association*, Portugal: Lisbon.

Biographical note

Stephan Schlosser (sschlos1@uni-goettingen.de) is a doctoral candidate and research associate at the Center of Methods in Social Sciences at the University of Göttingen, Germany. His research focuses on web-survey design and paradata/sensor data.

Jan Karem Höhne (hoehne@uni-mannheim.de) is a postdoctoral researcher at the Collaborative Research Center SFB 884 "Political Economy of Reforms" at the University of Mannheim, Germany. His research focuses on survey methodology, web-survey design, paradata/sensor data, and eye tracking.

Acknowledgment

The authors are grateful to Dagmar Krebs (University of Gießen) and Steffen Kühnel (University of Göttingen) for their great support. We also would like to thank Tanja Kunz (GESIS – Leibniz Institute for the Social Sciences) and Anke Metzler (Darmstadt University of Technology) for their excellent recommendations and for testing the application of the ECSP paradata codes.

Appendix A

In the following, we share the ECSP JavaScript codes from 2016 and 2018. Before using ECSP, we highly recommend reading the chapters “ECSP and ethical considerations” and “ECSP disclaimer” above. We listed all codes in an alphabetical order, except for response and start times because they need to be collected first.

```
{*EMBEDDED CLIENT SIDE PARADATA (Schlosser & Höhne, 2018) - JAVASCRIPT
CODE*}


<html>
<head>{$sys_head}

{* JQUERY *}
<script type='text/javascript'
src="//ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">'></script
>

</head>

<title>{$msg_title}</title>
<script type='text/javascript'>

{literal}

jQuery.noConflict();


//RESPONSE TIME (1)

var now=(function() {
  var performance=window.performance || {};
    performance.now=(function() {
    return performance.now    ||
    performance.webkitNow     ||
    performance.msNow         ||
    performance.oNow          ||
    performance.mozNow        ||
    function() {return new Date().getTime();};
  })();
  return performance.now();
});
var start=now();
x=new Date();


//CONNECTION TYPE

function connection_type() {
    "use strict";
    var strOnError, strConnection, strOut;
    strOnError="N/A";
    strConnection=null;
    strOut=null;
    try {
```

```
      strConnection=navigator.connection.type;
        strOut=strConnection;
    } catch(err) {
      return strOnError;
    }
    return strOut;
}
var ConnectionType;

(function($) {
$(document).ready(function() {
  ConnectionType=connection_type();
});
})(jQuery);


//DEVICE ORIENTATION AND ORIENTATION CHANGE

var ScreenTArray=new Array();
var ScreenOrientArray=new Array();

var mo={
    _browser: null,
    _os: null,
    _ua: navigator.userAgent,
    normalise: false,
    orientation: false,
    motion: false,
    init: function() {
        var orientation=false;
        var motion=false
        if (window.DeviceOrientationEvent) {
            orientation=true;
        }
        if (window.DeviceMotionEvent) {
            motion=true;
        }
        if (orientation && motion) {
            if (this._ua.match(/Firefox/i) && this._ua.match(/Android/i)) {
                this._os="Android";
                this._browser="Firefox";
            } else if (this._ua.match(/Android/i)) {
                this._os="Android";
                this._browser="Stock";
            } else if (this._ua.match(/Blackberry|RIM/i)) {
                this._os="Blackberry";
                this._browser="Stock";
            } else {
                this._os="iOS";
                this._browser="webkit";
            }
        } else if (orientation && !motion) {
            this._browser="Chrome";
            if (this._ua.match(/Android/i)) {
                this._os="Android";
            } else {
                this._os="Desktop";
            }
        } else if (!orientation) {
            this._browser="Unknown";
            this._os="Unknown";
        }
```

```
        this.orientation=orientation;
        this.motion=motion;
    },
} ;

(function($) {
function onresize2 () {
    ScreenTArray[ScreenTArray.length]=Math.round(now()-start);
    ScreenOrientArray[ScreenOrientArray.length]=window.orientation;
};
  $(window).resize(onresize2);
onresize2();
})(jQuery);


//KEY STROKES

var KeyTArray=new Array();
var KeyCodeArray=new Array();
var event='';

(function($) {
document.onkeypress=function(k) {
    var k=window.event || k;
    KeyCodeArray[KeyCodeArray.length]=(k.which || k.keyCode);
    KeyTArray[KeyTArray.length]=Math.round(now()-start);
   };
})(jQuery);


//MOUSE CLICK AND FINGER TAB

var ClickXArray=new Array();
var ClickYArray=new Array();
var ClickTArray=new Array();
var ClickXtext='';
var ClickYtext='';
var ClickTtext='';

(function($) {
  $(function() {
    $(document.body).mousedown(function(f) {
        ClickXArray[ClickXArray.length]=Math.round(f.pageX);
        ClickYArray[ClickYArray.length]=Math.round(f.pageY);
        ClickTArray[ClickTArray.length]=Math.round(now()-start);
    });
});
})(jQuery);


//MOUSE MOVEMENT

var XArray=new Array();
var YArray=new Array();
var TArray=new Array();
var Xtext='';
var Ytext='';
var Ttext='';

(function($) {
$(document).ready(function() {
    $(document.body).mousemove(function (e) {
```

```
        if (e.pageX !== 0 && e.pageY !== 0) {
            XArray[XArray.length]=Math.round(e.pageX);
            YArray[YArray.length]=Math.round(e.pageY);
            TArray[TArray.length]=Math.round(now()-start);
        }
    });
});
})(jQuery);


//SCREEN RESOLUTION AND PIXEL RATIO

var ScreenWArray;
var ScreenHArray;
var ScreenRatioArray;

ScreenWArray=screen.width;
ScreenHArray=screen.height;
ScreenRatioArray=window.devicePixelRatio || 1;


//SCROLLING (HORIZONTAL)

var ScrollHTArray=new Array();
var ScrollHXArray=new Array();
var scrollH='';

(function($) {
$(window).scroll(function() {
    var scrollH=$(document).scrollLeft();
    ScrollHXArray[ScrollHXArray.length]=scrollH;
    ScrollHTArray[ScrollHTArray.length]=Math.round(now()-start);
});
})(jQuery);


//SCROLLING (VERTICAL)

var ScrollVTArray=new Array();
var ScrollVYArray=new Array();
var scrollV='';

(function($) {
$(window).scroll(function() {
    var scrollV=$(document).scrollTop();
    ScrollVYArray[ScrollVYArray.length]=scrollV;
    ScrollVTArray[ScrollVTArray.length]=Math.round(now()-start);
});
})(jQuery);


//BASIC SURVEYFOCUS (SF)

var FocusTArray=new Array();
var FocusArray=new Array();
var state='';

var visibilityChange=(function(window) {
    var inView=true;
    return function(fn) {
window.onfocus=window.onblur=window.onpageshow=window.onpagehide=function(s
) {
```

```
            if ({focus:1, pageshow:1}[s.type]) {
                if (inView) return;
                fn("1"); //visible
                inView = true;
            } else if (inView) {
                fn("0"); //hidden
                inView = false;
            }
        };
    };
}(this));

visibilityChange(function(state) {
    FocusArray[FocusArray.length]=state;
    FocusTArray[FocusTArray.length]=Math.round(now()-start);
});


//MOBILE SURVEYFOCUS (SF)

var FocusTArray2=new Array();
var FocusArray2=new Array();
var state2='';
var mobout=false;

var FocusCheckr=function() {
    var _that={},
        _lastSeen=0,
        _timerInterval=10,
        _handlers=[];
        _that.timerThreshold=10;
        _that.onFocus=function(handler, params) {
        var hiddenProp=getHiddenProp();
        if (hiddenProp) {
            var evtName=hiddenProp.replace(/[H|h]idden/, "") +
"visibilitychange";
            document.addEventListener(evtName, function(e) {
                if (isHidden()) {
                    handler(e, params);
                }
            }, false);
        }else {
            var handlerObj={"handler": handler};
            if (params !== undefined) {handlerObj.params=params}
            _handlers.push(handlerObj);
            startLoop();
        }
    };
    _that.offFocus=function(handler, params) {
        var hiddenProp=getHiddenProp();
        if (hiddenProp) {
            var evtName=hiddenProp.replace(/[H|h]idden/, "") +
"visibilitychange";
            document.addEventListener(evtName, function(e) {
                if (!isHidden()) {
                    handler(e, params);
                }
            }, false);
        }
    };
    var animFrame=(function(req, can) {
        var af={},
```

```
    lastTime=0,
    vendors=["webkit", "moz", "o", "ms"];
    af[req]=window.requestAnimationFrame;
    af[can]=window.cancelAnimationFrame;
    for(var x=0, l=vendors.length; x < l && !af[req]; ++x) {
        af[req]=window[vendors[x]+'RequestAnimationFrame'];
        af[can]=window[vendors[x]+"CancelAnimationFrame"] ||
            window[vendors[x]+"CancelRequestAnimationFrame"];
    }
    if (!af[req]) {
        af[req]=function(callback, element) {
            var currTime=Date.now(),
                timeToCall=Math.max(0, 16-(currTime-lastTime)),
                id=window.setTimeout(function() {
                        callback(currTime+timeToCall);
                    }, timeToCall);
            lastTime=currTime+timeToCall;
            return id;
        };
    }
    if (!af[can]) {
        af[can]=function(id) {
            clearTimeout(id);
        };
    }
    return af;
}("request", "cancel"));
getHiddenProp=function() {
    var prefixes=["webkit", "moz", "o", "ms"],
        prop="";
    if ("hidden" in document) {return "hidden"}
    for (var i = 0, l = prefixes.length; i < l; i++) {
        prop=prefixes[i] + "Hidden";
        if ((prop) in document) {return prop};
    }
    return null;
},
isHidden=function() {
    return document[getHiddenProp()] || false;
},
startLoop=function() {
    _lastSeen=Date.now();
    window.onscroll=onScrollHandler;
    animFrame.request(rafHandler);
},
checkFocus=function() {
    if (Date.now()- _lastSeen > _that.timerThreshold) {
        notifyHandlers();
    }
    _lastSeen=Date.now();
},
rafHandler=function() {
    animFrame.request(rafHandler);
    checkFocus();
},
notifyHandlers=function() {
    var numHandlers=_handlers.length,
        handlerObj=null;
    if (numHandlers) {
        for (var i=0; i < numHandlers; i++) {
            handlerObj=_handlers[i];
            if (handlerObj["params"]) {
```

```
                    handlerObj.handler(null, handlerObj["params"]);
                }else {
                    handlerObj.handler(null);
                }
            }
        }
    },
    onScrollHandler=function(e) {
        _lastSeen=Date.now();
    };
    return _that;
};

var focusCheck=new FocusCheckr(), count=0;
focusCheck.timerThreshold=10;

focusCheck.onFocus(function() {
  FocusTArray2[FocusTArray2.length]=Math.round(now()-start);
  FocusArray2[FocusArray2.length]="0";
  mobout=true;
});

focusCheck.offFocus(function() {
    if(mobout==true) {
    FocusTArray2[FocusTArray2.length]=Math.round(now()-start);
    FocusArray2[FocusArray2.length]="1";
    mobout=false;
    }
});


//USER AGENT STRING

var UserAgentJS;
UserAgentJS=window.navigator.userAgent;


//WINDOW SIZE AND ZOOM

var SizeXArray=new Array();
var SizeYArray=new Array();
var SizeTArray=new Array();
var SizeXtext='';
var SizeYtext='';
var SizeTtext='';
var ZoomArray=new Array();
var Zoomtext='';

(function($) {
function onresize() {
    var Zoomtext=((window.outerWidth-16)/window.innerWidth);
    SizeXArray[SizeXArray.length]=$(window).width();
    SizeYArray[SizeYArray.length]=$(window).height();
    SizeTArray[SizeTArray.length]=Math.round(now()-start);
    ZoomArray[ZoomArray.length]=Zoomtext;
};
  $(window).resize(onresize);
onresize ();
})(jQuery);
```

```
///FUNCTION SEND / "NEXT" BUTTON

function send() {
y=new Date();

//RESPONSE TIME (2)

diff=Math.round(now()-start)


///CONNECTING VARIABLES WITH DATASET

//START TIME
MessStart=x.getTime();
var ZeitStart=document.forms[0].varZeitStart.value;
var ZeitStart_storage=eval("document.forms[0]."+ZeitStart);
ZeitStart_storage.value=MessStart;

//RESPONSE TIME
var ZeitDiff=document.forms[0].varZeitDiff.value;
var ZeitDiff_storage=eval("document.forms[0]."+ZeitDiff);
ZeitDiff_storage.value=diff;

//CONNECTION TYPE
var ConnectionTypeOut=document.forms[0].varConnectionTypeOut.value;
var ConnectionType_storage=eval("document.forms[0]."+ConnectionTypeOut);
ConnectionType_storage.value=ConnectionType;

//DEVICE ORIENTATION AND ORIENTATION CHANGE
var ScreenOrientArrayOut=document.forms[0].varScreenOrientArrayOut.value;
var
ScreenOrientArrayOut_storage=eval("document.forms[0]."+ScreenOrientArrayOut
);
ScreenOrientArrayOut_storage.value=ScreenOrientArray;

//DEVICE ORIENTATION AND CHANGE TIME
var ScreenTArrayOut=document.forms[0].varScreenTArrayOut.value;
var ScreenTArrayOut_storage=eval("document.forms[0]."+ScreenTArrayOut);
ScreenTArrayOut_storage.value=ScreenTArray;

//KEY STROKES TIME
var KeyTArrayOut=document.forms[0].varKeyTArrayOut.value;
var KeyTArrayOut_storage=eval("document.forms[0]."+KeyTArrayOut);
KeyTArrayOut_storage.value=KeyTArray;

//KEY STROKES
var KeyCodeArrayOut=document.forms[0].varKeyCodeArrayOut.value;
var KeyCodeArrayOut_storage=eval("document.forms[0]."+KeyCodeArrayOut);
KeyCodeArrayOut_storage.value=KeyCodeArray;

//MOUSE CLICK AND FINGER TAB X
var ClickXArrayOut=document.forms[0].varClickXArrayOut.value;
var ClickXArrayOut_storage=eval("document.forms[0]."+ClickXArrayOut);
ClickXArrayOut_storage.value=ClickXArray;

//MOUSE CLICK AND FINGER TAB Y
var ClickYArrayOut=document.forms[0].varClickYArrayOut.value;
var ClickYArrayOut_storage=eval("document.forms[0]."+ClickYArrayOut);
ClickYArrayOut_storage.value=ClickYArray;

//MOUSE CLICK AND FINGER TAB TIME
var ClickTArrayOut=document.forms[0].varClickTArrayOut.value;
```

```
var ClickTArrayOut_storage=eval("document.forms[0]."+ClickTArrayOut);
ClickTArrayOut_storage.value=ClickTArray;

//MOUSE MOVEMENT X
var XArrayOut=document.forms[0].varXArrayOut.value;
var XArrayOut_storage=eval("document.forms[0]."+XArrayOut);
XArrayOut_storage.value=XArray;

//MOUSE MOVEMENT Y
var YArrayOut=document.forms[0].varYArrayOut.value;
var YArrayOut_storage=eval("document.forms[0]."+YArrayOut);
YArrayOut_storage.value=YArray;

//MOUSE MOVEMENT TIME
var TArrayOut=document.forms[0].varTArrayOut.value;
var TArrayOut_storage=eval("document.forms[0]."+TArrayOut);
TArrayOut_storage.value=TArray;

//SCREEN RESOLUTION X
var ScreenWArrayOut=document.forms[0].varScreenWArrayOut.value;
var ScreenWArrayOut_storage=eval("document.forms[0]."+ScreenWArrayOut);
ScreenWArrayOut_storage.value=ScreenWArray;

// SCREEN RESOLUTION Y
var ScreenHArrayOut=document.forms[0].varScreenHArrayOut.value;
var ScreenHArrayOut_storage=eval("document.forms[0]."+ScreenHArrayOut);
ScreenHArrayOut_storage.value=ScreenHArray;

//SCREEN RESOLUTION PIXEL RATIO
var ScreenRatioArrayOut=document.forms[0].varScreenRatioArrayOut.value;
var
ScreenRatioArrayOut_storage=eval("document.forms[0]."+ScreenRatioArrayOut);
ScreenRatioArrayOut_storage.value=ScreenRatioArray;

//SCROLLING (HORIZONTAL) X
var ScrollHXArrayOut=document.forms[0].varScrollHXArrayOut.value;
var ScrollHXArrayOut_storage=eval("document.forms[0]."+ScrollHXArrayOut);
ScrollHXArrayOut_storage.value=ScrollHXArray;

//SCROLLING (HORIZONTAL) TIME
var ScrollHTArrayOut=document.forms[0].varScrollHTArrayOut.value;
var ScrollHTArrayOut_storage=eval("document.forms[0]."+ScrollHTArrayOut);
ScrollHTArrayOut_storage.value=ScrollHTArray;

//SCROLLING (VERTICAL) Y
var ScrollVYArrayOut=document.forms[0].varScrollVYArrayOut.value;
var ScrollVYArrayOut_storage=eval("document.forms[0]."+ScrollVYArrayOut);
ScrollVYArrayOut_storage.value=ScrollVYArray;

//SCROLLING (VERTICAL) TIME
var ScrollVTArrayOut=document.forms[0].varScrollVTArrayOut.value;
var ScrollVTArrayOut_storage=eval("document.forms[0]."+ScrollVTArrayOut);
ScrollVTArrayOut_storage.value=ScrollVTArray;

//BASIC SURVEYFOCUS (SF)
var FocusArrayOut=document.forms[0].varFocusArrayOut.value;
var FocusArrayOut_storage=eval("document.forms[0]."+FocusArrayOut);
FocusArrayOut_storage.value=FocusArray;

//BASIC SUVEYFOCUS (SF) TIME
var FocusTArrayOut=document.forms[0].varFocusTArrayOut.value;
var FocusTArrayOut_storage=eval("document.forms[0]."+FocusTArrayOut);
```

```
FocusTArrayOut_storage.value=FocusTArray;

//MOBILE SURVEYFOCUS (SF)
var FocusArrayOut2=document.forms[0].varFocusArrayOut2.value;
var FocusArrayOut2_storage=eval("document.forms[0]."+FocusArrayOut2);
FocusArrayOut2_storage.value=FocusArray2;

//MOBILE SURVEYFOCUS (SF) TIME
var FocusTArrayOut2=document.forms[0].varFocusTArrayOut2.value;
var FocusTArrayOut2_storage=eval("document.forms[0]."+FocusTArrayOut2);
FocusTArrayOut2_storage.value=FocusTArray2;

//USER AGENT STRING
var UserAgentJSOut=document.forms[0].varUserAgentJSOut.value;
var UserAgentJSOut_storage=eval("document.forms[0]."+UserAgentJSOut);
UserAgentJSOut_storage.value=UserAgentJS;

//WINDOW SIZE X
var SizeXArrayOut=document.forms[0].varSizeXArrayOut.value;
var SizeXArrayOut_storage=eval("document.forms[0]."+SizeXArrayOut);
SizeXArrayOut_storage.value=SizeXArray;

//WINDOW SIZE Y
var SizeYArrayOut=document.forms[0].varSizeYArrayOut.value;
var SizeYArrayOut_storage=eval("document.forms[0]."+SizeYArrayOut);
SizeYArrayOut_storage.value=SizeYArray;

//WINDOW SIZE TIME
var SizeTArrayOut=document.forms[0].varSizeTArrayOut.value;
var SizeTArrayOut_storage=eval("document.forms[0]."+SizeTArrayOut);
SizeTArrayOut_storage.value=SizeTArray;

//ZOOM
var ZoomArrayOut=document.forms[0].varZoomArrayOut.value;
var ZoomArrayOut_storage=eval("document.forms[0]."+ZoomArrayOut);
ZoomArrayOut_storage.value=ZoomArray;

}

</script>

{/literal}

{* ECSP END *}
```

Appendix B

In the following, we share the ECSP HTML codes from 2016 and 2018. Before using ECSP, we highly recommend reading the chapters “ECSP and ethical considerations” and “ECSP disclaimer” above. We listed all codes in an alphabetical order, except for response and start times because they need to be collected first.

```
<!--EMBEDDED CLIENT SIDE PARADATA (Schlosser & Höhne, 2018) - HTML CODE-->


<!--START TIME-->
<input type=hidden name="varZeitStart" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--RESPONSE TIME-->
<input type=hidden name="varZeitDiff" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--CONNECTION TYPE-->
<input type=hidden name="varConnectionTypeOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--DEVICE ORIENTATION AND CHANGE-->
<input type=hidden name="varScreenOrientArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--DEVICE ORIENTATION AND CHANGE TIME-->
<input type=hidden name="varScreenTArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--KEY STROKE-->
<input type=hidden name="varKeyCodeArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--KEY STROKE TIME-->
<input type=hidden name="varKeyTArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--MOUSE CLICK AND FINGER TAB X-->
<input type=hidden name="varClickXArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--MOUSE CLICK AND FINGER TAB Y-->
<input type=hidden name="varClickYArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--MOUSE CLICK AND FINGER TAB TIME-->
<input type=hidden name="varClickTArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--MOUSE MOVEMENT X-->
<input type=hidden name="varXArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--MOUSE MOVEMENT Y-->
<input type=hidden name="varYArrayOut" value="v_XYZ">
```

```
<input type="hidden" name="v_XYZ" value="">

<!--MOUSE MOVEMENT TIME-->
<input type=hidden name="varTArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--SCREEN RESOLUTION X-->
<input type=hidden name="varScreenWArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--SCREEN RESOLUTION Y-->
<input type=hidden name="varScreenHArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--SCREEN RESOLUTION PIXEL RATIO-->
<input type=hidden name="varScreenRatioArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--SCROLLING (HORIZONTAL) X-->
<input type=hidden name="varScrollHXArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--SCROLLING (HORIZONTAL) TIME-->
<input type=hidden name="varScrollHTArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--SCROLLING (VERTICAL) Y-->
<input type=hidden name="varScrollVYArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--SCROLLING (VERTICAL) TIME-->
<input type=hidden name="varScrollVTArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--BASIC SURVEYFOCUS (SF)-->
<input type=hidden name="varFocusArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--BASIC SURVEYFOCUS (SF) TIME-->
<input type=hidden name="varFocusTArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--MOBILE SURVEYFOCUS (SF)-->
<input type=hidden name="varFocusArrayOut2" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--MOBILE SURVEYFOCUS (SF) TIME-->
<input type=hidden name="varFocusTArrayOut2" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--USER AGENT STRING-->
<input type=hidden name="varUserAgentJSOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--WINDOW SIZE X-->
<input type=hidden name="varSizeXArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--WINDOW SIZE Y-->
<input type=hidden name="varSizeYArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">
```

```
<!--WINDOW SIZE TIME-->
<input type=hidden name="varSizeTArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">

<!--ZOOM-->
<input type=hidden name="varZoomArrayOut" value="v_XYZ">
<input type="hidden" name="v_XYZ" value="">
```