Scott van Looy

Location:	Berlin, Germany (Permanent residency)
Nationality:	British
Professional IT experience since	1995
Year of Birth:	1974
Tel:	+49 1511 269 1574
Email:	scott@ethosuk.net

Summary

Scott is a specialist in front end web technologies and implementation. He prefers leading by example and as such is a very hands-on coder, believing that the quality of the product is paramount and happy to share his knowledge with the team.

He is an active participant in peer reviews and cross browser testing.

An expert in Javascript, having written his own libraries and frameworks in the past, he has recently been specialising in full stack web development using React/NextJS/Typescript.

Scott has worked primarily with the following technologies: Typescript, NextJS, React, SASS, Jest, Playwright, Javascript, NodeJS, HTML5 and CSS3 and as a consequence has also worked with PostCSS, Node, MongoDB, Express, Gulp, AWS, Google Cloud.

Scott can be found at Github and on Twitter.

Expert in

- Typescript 3+ years
- React 4+ years
- GraphQL 4+ years
- javascript 25+ years
- CSS 25+ years
- HTML 25+ years
- JSON 15+ years

- Playwright 1+ year
- NodeJS 9+ years
- SASS 10+ years
- Jest/RTL 5+ years
- Handlebars 5+ years

Additionally

- Adobe Creative Cloud 10+ years
- SQL Server (+MariaDB) 10 years
- JSP 10 years
- AEM 2 years

- EJS 10+ years
- Devops 10+ years
- Gulp 5+ years
- MongoDB 5+ years
- CouchDB 1 year
- Adobe CQ 1 year

Recent Contract Work

Nando's 2020/06 - 2024/05 Nando's Online Ordering

Senior Software Engineer working on the Online Ordering Platform. As a founding member of the Web Experience team at Nando's, Scott was instrumental in building the solid foundations of what would become a very successful e-commerce platform for Nando's. Choosing NextJS for its flexibility, Scott and the team built a world class onine ordering experience that's giving joy to Nando's customers across the UK and Ireland for eat-in, collection and delivery orders.

Technologies used: React, Typescript, NextJS, graphQL, OpenAI ChatGPT, MobX, MST, Jest, Jest-aXe, Playwright, Cypress, Github, SASS modules, Atlassian, Slack

Kenza (Contract - solo) 2018/08 - present kenza.io, also LKQ Europe

Head of technology. Built both of Kenza's corporate websites to date as well as various sites for LKQ Europe and their subsidiaries. Also built SAP Leonardo microsite advertising their service.

Technologies used: React, Acquia SiteStudio, Javascript, PHP, Gulp, Wordpress, Drupal, HTML5 Canvas, CSS3, Video API, Sound API, WebGL, FFmpeg.

AKQA Berlin (Contract) 2020/04 Portugieser 2020 Collection (archive.org link) - unfortunately no longer live, the archive.org version is missing many images.

Team lead on the IWC Schaffhausen SPA microsite. Very tight deadline and had to work flawlessly in markets including China.

Technologies used: React, Babel, Netlify, Cloudinary, Gatsby, Github.

Primer SF (Contract) 2019/07 - 2019/09

Working on B2B AI tool for Primer clients. Info about Primer: primer.ai

Technologies used: React, Typescript, MobX State Tree, ES6, Github, Greensock.

Sportsvertise (solo) 2018/08 - 2020/02 sportsvertise.com

Senior Engineer. Creating SPA for online auction site dealing with selling of sports advertising space.

Technologies used: HTML5, CSS3, Vanilla JS, HTML5 Video, Web components.

Submerge (solo) Various between 2018/09 - present submerge.me

Festival and corporate website. Implementing redesign.

Technologies used: HTML5, CSS3, Vanilla JS, Video, Wordpress.

Siemens via AKQA Berlin 2018/04 - 2018/06

Creating HTML5 Canvas animation for an app

Technologies used: Gulp, HTML5 Canvas, vanilla Javascript.

Also of note

SAP via TLGG - microsite advertising SAP Leonardo - 2018/01 - 2018/04

Zalando - Skunkworks team building vanilla JS front end - 2017/04 - 2017/12

Hermès via AKQA - Flagship Drupal/Magento eCommerce site for Hèrmes - 2016/06 - 2017/04 hermes.com

Submerge Festival - Rebuilding site - 2016/08 - 2016/11 Submerge Festival

Zalando via Futurice - Building Zalando Jobs - 2016/04

The Legacy Section

Before taking up contracting, Scott spent 9 years as an employee working at AKQA, 4 years as a Technical Architect/Development lead and 5 years as a Senior Web Developer. In this time he has worked on many projects including:

An as yet unnamed community management tool 2015/12 - 2016/3

Architect/Full stack developer - Integrating with IBM's Watson service, this Node based tool allows Community Managers to work in a much more efficient fashion on Twitter. Taking advantage of IBM's Natural Language Processor and Natural Language Classifier, their Personality Insights service and Sentiment Analysis to offer a semi automatic conversation, enabling CMs to interact with many more end users than previously able.

Technologies used: AI, HTML5, CSS3, Javascript (ES5 front end, ES6 back end), NodeJS, Express, Passport, jQuery, Handlebars, MongoDB, Twitter API, IBM Bluemix APIs (NLP, NLC, Sentiment Analysis, PI).

Rolls-Royce Motor Cars 2015/06 - 2015/10 Rolls-Royce Motor Cars

Architect/Front end lead on Rolls-Royce Motorcars - Leading a front-end development team of 5 based in Germany/Ukraine, this was a pretty simple site without much in the way of complex user interactions, the key here was that the site must be smooth and beautiful and fast and feel premium. Working to aggressive deadlines, the final website launch was tied to the launch of a new car, the Rolls-Royce Dawn.

Using the Babel ES6 transpiler, the majority of the front end code was written in ES6, including our Gulp scripts. Scott developed a Gulp based way of working with Adobe CQ5 that allowed the team to iterate quickly and develop in real time with designers. Taking advantage of the open source PubSubJS library, loosely coupled front end components were created that used the publish/subscribe pattern to communicate with each other and user interactions.

Performance was key, so using the latest draft picture spec and the picturefill polyfill, adaptive images were created that could download incredibly fast as well as look beautiful on large, high resolution screens. Using progressive enhancement techniques, the site is optimised for SEO and a non-javascript experience. Gzipped and minified single CSS files, graphics as compressed SVGs whereever possible, all to make the site feel fast as well as premium quality.

Project came in on time and under budget. Launched in September 2015.

Technologies used: Gulp, ECMAScript 6, HTML5, jQuery, Picturefill, SVG, PubSubJS, SASS, Babel, ZombieJS, Adobe CQ5 version 6, Sightly, Stash, Slack, Java.

Officine Panerai 2014/01 - 2015/06 Panerai

Front end lead on Panerai - Leading a front-end development team of 5 based in Germany/India, this was a pretty simple site showcasing the work of the Italian watch designers. Customising google maps to give an on brand boutique browsing experience and focussing on front end performance and making the interface smooth and simple to use. Built on Adobe CQ5 version 5. Using the open source jQuery library, the team developed a number of custom jQuery plugins to augment the user experience.

Included 2 weeks of training existing front end developers and interviewing front end candidates in Gurgaon, India.

Technologies used: HTML5, CSS3, ECMAScript 5. Browser support included all major vendors and even IE8. Optimised for mobile performance.

Volkswagen Motorcars 2010/07 - 2012/06 Video about the iHDCC Car Configurator

Front end lead on the VW configurator: Leading a front end development team of 4 based in London. This is a single page app, Scott wrote a Javascript based MVC framework to power the front end. The configurator is powered by realtime factory inventory data so configuring and accessorizing a car is entirely possible online, but the client didn't want a "buy" button as they assumed their relationships with dealers would suffer.

Many weakly coupled components on the page communicated changes to the car configuration via an event based system. Real time data came from VW's own back end systems via SOAP which was converted as-is to JSON for the front end to consume.

Version 1 was launched at the start of October 2010 in the US market as a static HTML website powered by JSON services. Version 2 was retooled to sit upon Liferay CMS for personalisation and went on to be launched in multiple markets in late 2012.

Technologies used: HTML, CSS, Javascript, Photoshop

Nokia 2009/02 - 2010/07

Web developer on the Ovi Maps core team for Nokia working with 7 others out of Nokia's Berlin office.

Scott started on this project as a web developer working in the scrum team under our scrum master. By the end of the project he was scrum master as well as lead developer.

Nokia had bought Navteq and decided to create an online mapping experience along the lines of Google maps. They had invested money in developing a browser plugin that would allow maps and navigation to work seamlessly in the browser.

A few months after Scott joined, it became clear that the existing event driven framework that ran the separate modules that made up the Ovi maps web experience wasn't performing very well once the team had moved away from the browser plugin and implemented the newly created Javascript maps API and tile maps. He proposed a radical solution to the then head of Ovi Maps - namely that the team be split in two. He suggested four of the team could spend two months building a brand new proof of concept Javascript framework, built for speed from the ground up and the rest of the team would continue to work on the existing framework in case the first team were delayed or failed. After one month of work and seeing the amazing speed of the first team had had written, the original project was canned and the "Cheetah" framework was born, everyone switching to work full time on the prototype.

This was an event driven MVC front end Javascript framework utilising prototypal inheritance and weakly coupled front end components communicating via an events object. The Javascript was then concatenated and minified using Google's Closure compiler and shipped as a single JS file.

Initially loading just enough Javascript to power the initial view, using the proxy pattern, the other modules could then be loaded in at runtime transparently to the views, with each view's code being downloaded and initialised on demand.

Technologies used: HTML, CSS, Javascript, JSON APIs, Nokia Maps JS API, Google Closure Compiler/optimiser/ JSDoc with closure hints for typing. Java.

Personal:

Scott is a passionate photographer, loves travelling with his camera and enjoys live music and cycling.

References:

Available on request