Supplementary material for
Detecting Locations in JavaScript Programs Affected by Breaking Library Changes

The following lists contain the detection patterns used in the experimental evaluation of TAPIR. Each item corresponds to one entry in one of the changelogs and shows the corresponding TAPIR patterns. The breaking change descriptions marked with '⋆' are the undocumented ones that we discovered while conducting the experiments.

lodash 4.0.0

(1) Made _#times, _#forEach, _#forIn, _#forOwn, & their right-forms implicitly end chain sequences.

```
• call <lodash>()**.{times,forEach,forEachRight,forIn
                     ,forInRight,forOwn,forOwnRight}().value [0, 0]
```

(2) Removed category names from module paths

```
• import lodash/{collection,number,chain,function
                ,math,array,date,lang,object,string,utility}/*
```

(3) Removed _.pluck in favor of _.map with iteratee shorthand

```
• read {<lodash>,<lodash/collection>
       ,<lodash>()**,<lodash>.chain()**}.pluck
• import lodash/collection/pluck
```

(4) Removed thisArg params from most methods because they were largely unused, complicated implementations, & can be tackled with _.bind, Function#bind, or arrow functions

```
• call {<lodash>,<lodash/utility>}.{callback,iteratee} [2, 2]
• call {<lodash>,<lodash/{collection,number,chain,function,math
                        ,array,date,lang,object,string,utility}>}
      .{dropRightWhile,dropWhile,findIndex,findLastIndex,remove
       ,takeRightWhile,takeWhile,unzipWith,zipWith,tap,thru
       ,countBy,every,all,filter,select,find,findLast,forEach
       ,each,forEachRight,forEachRight,groupBy,indexBy,map
       ,collect,partition,reject,some,any,sortBy,cloneDeep
       ,max,min,sum,findKey,findLastKey,forIn,forInRight
       ,forOwn,forOwnRight,mapKeys,mapValues,omit,pick,times}
      [3, 3] 1:function
• call {<lodash>(),<lodash>.chain()}**
      .{dropRightWhile,dropWhile,findIndex,findLastIndex
       ,remove,takeRightWhile,takeWhile,unzipWith,zipWith
       ,tap,thru,countBy  ,every,all,filter,select,find
       ,findLast,forEach,each,forEachRight,forEachRight
       ,groupBy,indexBy,map,collect,partition,reject
       ,some,any,sortBy,cloneDeep,max,min,sum,findKey
       ,findLastKey,forIn,forInRight,forOwn,forOwnRight
       ,mapKeys,mapValues,omit,pick,times} [2, 2] 0:function
• call <lodash/{collection,number,chain,function,math,array,date
               ,lang,object,string,utility}/{dropRightWhile
               ,dropWhile,findIndex,findLastIndex,remove
               ,takeRightWhile,takeWhile,unzipWith,zipWith,tap
               ,thru,countBy,every,all,filter,select,find,findLast
               ,forEach,each,forEachRight,groupBy,indexBy,map
               ,collect,partition,reject,some,any,sortBy
               ,cloneDeep,max,min,sum,findKey,findLastKey
               ,forIn,forInRight,forOwn,forOwnRight,mapKeys
               ,mapValues,omit,pick,times}> [3, 3] 1:function}
• call <lodash/utility/{callback,iteratee}> [2, 2]}
• call {<lodash>,<lodash/{collection,number,chain,function,math
                        ,array,date,lang,object,string,utility}>}
       .{hsortedIndex,sortedLastIndex,uniq,unique,reduce,foldl
```

```
          ,reduceRight,foldr,clone,isEqual,eq,isMatch,transform} [4, 4]}
    • call {<lodash>,<lodash/{collection,number,chain,function,math
                               ,array,date,lang,object,string,utility}>}
           .{assign,extend,merge} [4,]}
    • call <lodash/{collection,number,chain,function,math,array
                     ,date,lang,object,string,utility}
           /{sortedIndex,sortedLastIndex,uniq,unique,reduce,foldl
            ,reduceRight,foldr,clone,isEqual,eq,isMatch,transform}>
           [4, 4]}
    • call <lodash/{collection,number,chain,function,math,array
                     ,date,lang,object,string,utility}
           /{assign,extend,merge}> [4,]}
```

(5) Split _.max & _.min into _.maxBy & _.minBy
```
    • call {<lodash>,<lodash/math>}.{max,min} [2, 3]
    • call <lodash/math/{min,max}> [2, 3]
```
(6) Removed _.support
```
    • read <lodash>.support
```
(7) Removed _.findWhere in favor of _.find with iteratee shorthand
```
    • read {<lodash>,<lodash/collections>}.findWhere
    • import lodash/collections/findWhere
```
(8) Removed _.where in favor of _.filter with iteratee shorthand
```
    • read {<lodash>,<lodash/collections>).where
    • import lodash/collections/where
```
(9) Renamed _.indexBy to _.keyBy
```
    • read {<lodash>,<lodash/collections>}.indexBy
    • import lodash/collections/indexBy
```
(10) Renamed _.invoke to _.invokeMap
```
    • read {<lodash>,<lodash/collections>}.invoke
    • import lodash/collections/invoke
```
(11) Renamed _.modArgs to _.overArgs
```
    • read {<lodash>,<lodash/function>}.modArgs
    • import lodash/function/modArgs
```
(12) Renamed _.padLeft & _.padRight to _.padStart & _.padEnd
```
    • read {<lodash>,<lodash/string>}.{padLeft,padRight}
    • import lodash/string/{padLeft,padRight}
```
(13) Renamed _.pairs to _.toPairs
```
    • read {<lodash>,<lodash/object>}.pairs
    • import lodash/object/pairs
```
(14) Renamed _.rest to _.tail
```
    • read {<lodash>,<lodash/array>}.rest
    • import lodash/array/rest
```
(15) Renamed _.restParam to _.rest
```
    • read {<lodash>,<lodash/function>}.restParam
    • import lodash/function/restParam
```
(16) Renamed _.sortByOrder to _.orderBy
```
    • read {<lodash>,<lodash/collections>}.sortByOrder
    • import lodash/collections/sortByOrder
```
(17) Renamed _.trimLeft & _.trimRight to _.trimStart & _.trimEnd
```
    • read {<lodash>,<lodash/string>}.{trimLeft,trimRight}
    • import lodash/string/{trimLeft,trimRight}
```
(18) Renamed _.trunc to _.truncate
```
    • read {<lodash>,<lodash/string>}.trunc
    • import lodash/string/trunc
```
(19) Split _.assign & _.assignIn into _.assignWith & _.assignInWith
```
    • call {<lodash>,<lodash/object>}.assign [3,]
    • call <lodash/object/assign> [3,]
```

(20) Split _.clone & _.cloneDeep into _.cloneWith & _.cloneDeepWith
- `call {<lodash>,<lodash/lang>}.cloneDeep [2, 3]`
- `call <lodash/lang/cloneDeep> [2, 3]`

(21) Split _.indexOf & _.lastIndexOf into _.sortedIndexOf & _.sortedLastIndexOf
- `call {<lodash>,<lodash/array>}.{indexOf,lastIndexOf} [3, 3] 2:boolean`
- `call <lodash/array/{indexOf,lastIndexOf}> [3, 3] 2:boolean`

(22) Split _.invert into _.invertBy (see v4.1.0)
- `call {<lodash>,<lodash/object>}.invert [2, 2]`
- `call <lodash/object/invert> [2, 2]`

(23) Split _.isEqual into _.isEqualWith
- `call {<lodash>,<lodash/lang>}.isEqual [3, 4]`
- `call <lodash/lang/isEqual> [3, 4]`

(24) Split _.isMatch into _.isMatchWith
- `call {<lodash>,<lodash/lang>}.isMatch [3, 4]`
- `call <lodash/lang/isMatch> [3, 4]`

(25) Split _.merge into _.mergeWith
- `call {<lodash>,<lodash/object>}.merge [3,]`
- `call <lodash/object/merge> [3,]`

(26) Split _.omit & _.pick into _.omitBy & _.pickBy
- `call {<lodash>,<lodash/object>}.{omit,pick} [2, 2] 1:function`
- `call <lodash/object/{omit,pick}> [2, 2] 1:function`

(27) Split _.sample into _.sampleSize
- `call {<lodash>,<lodash/collections>}.sample [2, 2]`
- `call <lodash/collections/sample> [2, 2]`

(28) Split _.sortedIndex into _.sortedIndexBy
- `call {<lodash>,<lodash/array>}.sortedIndex [3, 4]`
- `call <lodash/array/sortedIndex> [3, 4]`

(29) Split _.sortedLastIndex into _.sortedLastIndexBy
- `call {<lodash>,<lodash/array>}.sortedLastIndex [3, 4]`
- `call <lodash/array/sortedLastIndex> [3, 4]`

(30) Split _.sum into _.sumBy
- `call {<lodash>,<lodash/math>}.sum [2, 3]`
- `call <lodash/math/sum> [2, 3]`

(31) Split _.uniq into _.sortedUniq, _.sortedUniqBy, & _.uniqBy
- `call {<lodash>,<lodash/array>}.uniq [2, 2] 1:boolean`
- `call {<lodash>,<lodash/array>}.uniq [2, 3] 1:function`
- `call {<lodash>,<lodash/array>}.uniq [3, 4]`
- `call <lodash/array/uniq> [2, 2] 1:boolean`
- `call <lodash/array/uniq> [2, 3] 1:function`
- `call <lodash/array/uniq> [3, 4]`

(32) Split _.zipObject into _.fromPairs
- `call {<lodash>,<lodash/array>}.zipObject [1, 1]`
- `call <lodash/array/zipObject> [1, 1]`

(33) Absorbed _.sortByAll into _.sortBy
- `call {<lodash>,<lodash/collections>}.sortByAll`
- `call <lodash/collections/sortByAll>`

(34) Changed the category of _.at to “Object”
- `read <lodash/collections>.at`
- `import lodash/collections/at`

(35) Changed the category of _.bindAll to “Util”
- `read <lodash/function>.bindAll`
- `import lodash/function/bindAll`

(36) Changed _.matchesProperty shorthand to an array of [path, srcValue]
- `call {<lodash>,<lodash/{collection,number,chain,function,math`

```
                          ,array,date,lang,object,string,utility}>}
          .{dropRightWhile,dropWhile,findIndex,findLastIndex,remove
           ,takeRightWhile,takeWhile,countBy,every,all
           ,filter,select,find,detect,groupBy,indexBy
           ,map,collect,partition,reject,some,any,sortBy
           ,max,min,findKey,findLastKey,mapValues,uniq,unique}
          [3, 3] 1:string
• call <lodash/{collection,number,chain,function,math,array
                ,date,lang,object,string,utility}
         /{dropRightWhile,dropWhile,findIndex,findLastIndex
          ,remove,takeRightWhile,takeWhile,countBy,every
          ,all,filter,select,find,detect,groupBy,indexBy
          ,map,collect,partition,reject,some,any,sortBy
          ,max,min,findKey,findLastKey,mapValues,uniq,unique}>
      [3, 3] 1:string
• call {<lodash>,<lodash/array>}.{sortedIndex,uniq,unique}
         [4, 4] 2:string
• call <lodash/array/{sortedIndex,uniq,unique}>
         [4, 4] 2:string
```

(37) Enabled _.merge to assign undefined if the destination property doesn't exist

```
• call {<lodash>,<lodash/object>}.merge
• call <lodash/object/merge>
```

(38) Made "By" methods like _.groupBy & _.sortBy provide a single param to iteratees

```
• call {<lodash>,<lodash/collections>}
         .{sortBy,countBy,groupBy,indexBy}
         [2, 3] 1:{function2,function3}
• call <lodash/collections/{sortBy,countBy,groupBy,indexBy}>
         [2, 3] 1:{function2,function3}
```

(39) Made _.add, _.max, _.min, & _.sum no longer coerce values to numbers

```
• call {{<lodash>,<lodash/math>}
         .{max,min,sum},<lodash/math/{max,min,sum}>} [2, 3]
• call {{<lodash>,<lodash/math>}
         .{max,min,sum},<lodash/math/{max,min,sum}>} [1, 1]
• call {{<lodash>,<lodash/math>}.add,<lodash/math/{add}>}
```

(40) Made _.capitalize uppercase the first character & lowercase the rest (see _.upperFirst)

```
• call {<lodash>,<lodash/string>}.capitalize [0, 1]
• call <lodash/string/capitalize> [0, 1]
```

(41) Made _.eq its own method instead of an alias for _.isEqual

```
• call {<lodash>,<lodash/lang>}.eq
• call <lodash/lang/eq>
```

(42) Made _.functions return only own method names

```
• call {<lodash>,<lodash/object>}.{methods,functions}
• call <lodash/object/functions>
```

(43) Made _.max & _.min return undefined when passed an empty array

```
• call {<lodash>,<lodash/math>}.max
• call {<lodash>,<lodash/math>}.min
• call <lodash/math/max>
• call <lodash/math/min>
```

(44) Made _.words chainable by default

```
• call <lodash>()**.words
```

(45) Removed isDeep params from _.clone & _.flatten

```
• call {<lodash>,<lodash/lang>}.clone [2, 4] 1:boolean
• call {<lodash>,<lodash/lang>}.clone [2, 3] 1:function
• call <lodash/lang/clone> [2, 4] 1:boolean
• call <lodash/lang/clone> [2, 3] 1:function
```

(46) Removed support for binding all methods by default from _.bindAll

```
• call {<lodash>,<lodash/function>}.bindAll [1, 1]
• call <lodash/function/bindAll> [1, 1]
```

(47) Dropped boolean options param support in _.debounce, _.mixin, & _.throttle
```
• call {<lodash>,<lodash/{function,utility}>}
        .{debounce,throttle,mixin} [3, 3] 2:boolean
• call <lodash/{function,utility}/{debounce,throttle,mixin}>
        [3, 3] 2:boolean
```
(48) Dropped support for boolean orders param in _.orderBy
```
• call {<lodash>,<lodash/collections>}.sortOrderBy [3, 3]
• call <lodash/collections/sortOrderBy> [3, 3]
```
(49) Made _.escapeRegExp align to the defunct ES7 proposal
```
• call {<lodash>,<lodash/string>}.escapeRegExp [1, 1]
• call <lodash/string/escapeRegExp> [1, 1]
```
(50) Made _.max, _.min, & _.sum support arrays only
```
• call {<lodash>,<lodash/math>}.{max,min,sum} 0:{string,object}
• call <lodash/math/{max,min,sum}> 0:{string,object}
```
(51) Removed 17 aliases
```
• read {<lodash>,<lodash/{collection,number,chain,function,math
                          ,array,date,lang,object,string,utility}>
                 ,<lodash>()**,<lodash>.chain()**}
        .{all,any,backflow,callback,collect,compose,contains,detect
         ,foldl,foldr,include,inject,methods,object,#run,select,unique}
• import lodash/{collection,number,chain,function,math,array
                 ,date,lang,object,string,utility}
           /{all,any,backflow,callback,collect,compose,contains
            ,detect,foldl,foldr,include,inject,methods,object
            ,#run,select,unique}
```
async 3.0.0

(1) In queue, priorityQueue, cargo and cargoQueue, the "event"-style methods, like q.drain and q.saturated are now methods that register a callback, rather than properties you assign a callback to. They are now of the form q.drain(callback). If you do not pass a callback a Promise will be returned for the next occurrence of the event, making them await-able, e.g. await q.drain(). (#1586, #1641)
```
• write {<async>.{queue,priorityQueue,cargo,cargoQueue}
         ,<async/{queue,priorityQueue,cargo,cargoQueue}>}()?
         .{drain,saturated,unsaturated,empty,error}
```
(2) during and doDuring have been removed, and instead whilst, doWhilst, until and doUntil now have asynchronous test functions. (#850, #1557)
```
• read <async>.{during,doDuring}
• import async/{during,doDuring}
• call (<async>.{whilst,until,doWhilst,doUntil}
        ,<async/{whilst,until,doWhilst,doUntil}>)
```
(3) memoize no longer memoizes errors (#1465, #1466)
```
• call {<async>.memoize,<async/memoize>}
```
(4) applyEach/applyEachSeries have a simpler interface, to make them more easily type-able. It always returns a function that takes in a single callback argument. If that callback is omitted, a promise is returned, making it awaitable. (#1228, #1640)
```
• call {<async>.{applyEach,applyEachSeries}
        ,<async/{applyEach,applyEachSeries}>}
        [2,]
```
express 4.0.0

(1) remove connect and connect's patches except for charset handling
```
• read <express>?**.headerSent
• call <express>?**.on [2, 2] 0:"header" 1:function
• call <express>?**.setHeader 0:"Set-Cookie" 1:{string,array}
```
(2) remove bodyParser middelware
```
• call <express>.{bodyParser,json,urlencoded}
```

(3) remove all bundled middleware except static
- `call <express>.{compress,timeout,cookieParser,cookieSession`
  `,csrf,errorHandler,session,methodOverride`
  `,logger,responseTime,favicon,directory,vhost}`

(4) remove express.createServer() - it has been deprecated for a long time. Use express()
- `read <express>.createServer`

(5) remove app.configure - use logic in your own app code
- `call <express>?**.configure [1, 1]`
- `call <express>?**.configure [2, 2]`

(6) remove app.router - is removed
- `read <express>?**.router`

(7) remove req.auth - use basic-auth instead
- `read <express>.basicAuth`

(8) remove req.accepted* - use req.accepts*() instead
- `read <express>?**.{acceptedCharsets,acceptedLanguages}`

(9) res.location - relative URL resolution is removed
- `call <express>?.location [1, 1] 0:string`

(10) remove res.charset - include the charset in the content type when using res.set()
- `write <express>?**.charset`

(11) change app.route -> app.mountpath when mounting an express app in another express app
- `read <express>()?**.route`

(12) change req.accepts* -> req.accepts*s - i.e. req.acceptsEncoding -> req.acceptsEncodings
- `read <express>?**.{acceptsEncoding,acceptsCharset,acceptsLanguage}`

(13) change req.params is now an object instead of an array
- `read <express>?**.params`

(14) change res.locals is no longer a function. It is a plain js object. Treat it as such.
- `call <express>?**.locals`

(15) refactor req.accepts* with accepts
- `call <express>?**.{acceptsCharset,acceptsLanguage}`

(16) refactor req.is with type-is
- `call <express>?**.is`

(17) ⋆Removed app.routes
- `read <express>?**.routes`

(18) ⋆express.mime is removed. Use express.static.mime
- `read <express>.mime`

chalk 2.0.0

(1) Removed chalk.hasColor(). Use the has-ansi package directly instead. 04cae22
- `read <chalk>.hasColor`

(2) Removed chalk.stripColor(). Use the strip-ansi package directly instead. 04cae22
- `read <chalk>.stripColor`

(3) Removed chalk.styles. Use the ansi-styles package directly instead. 8702496
- `read <chalk>.styles`

bluebird 3.0.0

(1) If a second argument is passed to Promise.promisify, Promise.promisifyAll or Promise.asCallback, it should be wrapped inside an object, i.e., if the second argument is arg2, then the patch should make the second argument into the object {context: arg2}.
- `call <bluebird>.{promisify,promisifyAll,asCallback} [2, 2]`

(2) Previously if the callback to be promisified was with multiple values the resolved value was an array off all the arguments and just the value in the case of the callback only being called with one argument. Now by default the resolved value is always a single even though the callback is called multiple times. There is however an option that can be enabled, such that

the resolved value always is an array (multiArgs). The patch could then use this option and in the case of a single element array, resolve with the single value.
- `call <bluebird>.{promisify,asCallback}`
- `call <bluebird>.promisifyAll`

(3) Cancellation redesign.
- `read <bluebird>?**.{cancellable,uncancellable}`

(4) Promise progression has been completely removed.
- `read <bluebird>?**.{progressed,progress}`
- `call <bluebird>?**.{then,done,fork} [3, 3]`

(5) .spread's second argument has been removed.
- `call <bluebird>?**.spread [2, 2]`

(6) .done causes an irrecoverable fatal error in Node.js environments now. See #471 for rationale.
- `call <bluebird>?**.done`

(7) Errors created with Promise.reject or reject callback of new Promise are no longer marked as OperationalErrors.
- `call <bluebird>?**.error [1, 1] 0:function`

(8) ⋆Submodules moved from folder 'main' to folder 'release'
- `import bluebird/js/main/*`

uuid 3.0.0

(1) remove .parse and .unparse
- `call <uuid>.{parse,unparse}`

commander 3.0.0

(1) Breaking custom event listeners: –no-foo on cli now emits option:no-foo (previously option:foo)
- `call <commander>.on [2, 2] 0:string 1:function`

(2) Change to use straight quotes around strings in error messages (like 'this' instead of ‘this')
- `call <commander>**.parse`

(3) Output format of –help changed
- `call <commander>**.parse`

rxjs 6.0.0

(1) webSocket: webSocket creator function now exported from rxjs/websocket as websocket.
- `import rxjs/**/*observable/dom/{WebSocket,WebSocketSubject}{,*,/**/*}`

(2) utils: Many internal use utilities like isArray are now hidden under rxjs/internal, they are implementation details and should not be used.
-
```
import rxjs/**/*util/{isArray,applyMixins,ArgumentOutOfRangeError
                     ,EmptyError,errorObject,identity,Immediate
                     ,isDate,isNumeric,isObject,isFunction,isPromise
                     ,isScheduler,noop,not,ObjectUnsubscribedError
                     ,pipe,root,SubscribeToResult,TimeoutError
                     ,toSubscriber,tryCatch,UnsubscriptionError}
       {,*,/**/*}
```

(3) testing observables: HotObservable and ColdObservable, and other testing support types are no longer exported directly.
- `import rxjs/**/*testing/{HotObservable,ColdObservable}{,*,/**/*}`

(4) creation functions: All create functions such as of, from, combineLatest and fromEvent should now be imported from rxjs/create.
-
```
read <rxjs{,/**/*}>.Observable
      .{from,fromPromise,of,combineLatest,fromEvent
       ,timer,interval,merge,bindCallback,bindNodeCallback
       ,empty,if,throw,defer,concat,combineAll,concatAll
       ,endWith,forkJoin,mergeAll,pairwise,race,startWith
       ,withLatestFrom,zip,ajax,generate,range}
```
- `import rxjs/observable/{from,fromPromise,of,combineLatest`

```
                        ,fromEvent,timer,interval,merge
                        ,bindCallback,bindNodeCallback,empty
                        ,if,throw,defer,concat,combineAll
                        ,concatAll,endWith,forkJoin,mergeAll
                        ,pairwise,race,startWith,withLatestFrom
                        ,zip,ajax,generate,range}
```

(5) symbols: Symbols are no longer exported directly from modules such as rxjs/symbol/observable please use Symbol.observable and Symbol.iterator (polyfills may be required)
- `import rxjs/symbol/*`
- `read <rxjs/symbol/*>.observable`

(6) deep imports: Can no longer deep import top-level types such as rxjs/Observable, rxjs/Subject, rxjs/ReplaySubject, et al. All imports should be done directly from rxjs, for example: import Observable, Subject
from 'rxjs';

-
  ```
  import rxjs/{Observable,Subject,ReplaySubject
              ,BehaviorSubject,Subscription,Subscriber}
  ```

(7) schedulers: Scheduler instances have changed names to be suffixed with Scheduler, (e.g. asap -> asapScheduler)
- `read <rxjs>.Scheduler.{async,virtualTime,animationFrame,asap,queue}`

(8) operators: Pipeable operators must now be imported from rxjs like so: import { map, filter, switchMap } from 'rxjs/operators';. No deep imports.

-
  ```
  import rxjs/operator/
         {audit,combineAll,debounceTime,do,findIndex,map,mergeScan
         ,publishBehavior,retry,single,switchMap,timeInterval
         ,windowTime,auditTime,combineLatest,defaultIfEmpty,elementAt
         ,find,mapTo,min,publish,retryWhen,skip,switchMapTo,timeout
         ,windowToggle,bufferCount,concatAll,delay,every,first
         ,materialize,multicast,publishLast,sample,skipLast,take
         ,timeoutWith,windowWhen,buffer,concat,delayWhen,exhaust
         ,groupBy,max,observeOn,publishReplay,sampleTime,skipUntil
         ,takeLast,timestamp,withLatestFrom,bufferTime,concatMap
         ,dematerialize,exhaustMap,ignoreElements,mergeAll
         ,onErrorResumeNext,race,scan,skipWhile,takeUntil,toArray
         ,zipAll,bufferToggle,concatMapTo,distinct,expand,isEmpty
         ,merge,pairwise,reduce,sequenceEqual,startWith,takeWhile
         ,toPromise,zip,bufferWhen,count,distinctUntilChanged
         ,filter,last,mergeMap,partition,repeat,share,subscribeOn
         ,throttle,windowCount,catch,debounce,distinctUntilKeyChanged
         ,finally,let,mergeMapTo,pluck,repeatWhen,shareReplay
         ,switch,throttleTime,window}
  ```

(9) ajax: Ajax observable should be imported from rxjs/ajax.
- `import rxjs/**/*observable/dom/ajax`

(10) Observable: You should no longer deep import custom Observable implementations such as ArrayObservable or ForkJoinObservable.
- `import rxjs/**/observable/EmptyObservable`
- `call <rxjs{,/**/*}>.EmptyObservable.create`

-
  ```
  import rxjs/**/observable
         /{ArrayLikeObservable,BoundNodeCallbackObservable
          ,DeferObservable,ForkJoinObservable,RangeObservable
          ,TimerObservable,ArrayObservable,FromObservable
          ,IfObservable,PairsObservable,ScalarObservable
          ,FromEventObservable,NeverObservable,PromiseObservable
          ,SubscribeOnObservable,UsingObservable
          ,ConnectableObservable,ErrorObservable,IntervalObservable
          ,BoundCallbackObservable,FromEventPatternObservable
          ,GenerateObservable,IteratorObservable}
  ```
- `read <rxjs/**/observable`

```
/{ArrayLikeObservable,FromObservable,IteratorObservable
 ,PromiseObservable,ArrayObservable,ScalarObservable
 ,BoundCallbackObservable,BoundNodeCallbackObservable
 ,DeferObservable,EmptyObservable,ErrorObservable
 ,ForkJoinObservable,FromEventObservable
 ,FromEventPatternObservable,GenerateObservable
 ,IfObservable,IntervalObservable,NeverObservable
 ,PairsObservable,RangeObservable,TimerObservable
 ,UsingObservable}>.create
```

(11) _throw: _throw is now exported as throwError
- `read <rxjs/**/observable/throw>._throw`

(12) if: if is now exported as iif
- `import rxjs/**/if`
- `read <rxjs>.Observable.if`

(13) operators removed: Operator versions of static observable creators such as merge, concat, zip, onErrorResumeNext, and race have been removed. Please use the static versions of those operations. e.g. a.pipe(concat(b, c)) becomes concat(a, b, c).
- `import rxjs/operator/{merge,concat,zip,onErrorResumeNext,race}{,/**/*}`
- `call (<rxjs{,/**/*}>?** \ <rxjs{,/**/*}>.Observable)`
  `.{merge,concat,zip,onErrorResumeNext,race}`

(14) Symbol.observable: RxJS will no longer be polyfilling Symbol.observable. That should be done by an actual polyfill library. This is to prevent duplication of code, and also to prevent having modules with side-effects in rxjs.
- `read <Symbol>.observable`

(15) Rx.ts: importing from rxjs/Rx is no longer available. Upcoming backwards compat solution will allow that
- `import rxjs/Rx`

(16) never: no longer exported. Use the NEVER constant instead.
- `import rxjs/**/never`

(17) ajax: will no longer execute a CORS request by default, you must opt-in with the crossDomain flag in the config.
- `call <rxjs{,/**/*}>.ajax 0:{string,object}`
- `call <rxjs/**/operator/ajax{,*,/**/*}> 0:{string,object}`

(18) websocket: WebSocketSubject will now JSON serialize all messages sent over it by default, to return to the old behavior, pass a config setting of serializer: x => x like so: websocket({ url, serializer: x => x })
- `call <rxjs/**/observable/dom/{webSocket,WebSocketSubject}{,*,/**/*}>`
  `0:string`
- `call <rxjs/**/observable/dom/{webSocket,WebSocketSubject}{,*,/**/*}>`
  `0:object`

(19) ⋆Removed fromPromise, should use from instead.
- `import rxjs/observable/fromPromise`
- `read <rxjs{,/**/*}>.Observable.fromPromise`

(20) ⋆Removed default import such that import Rx from 'rxjs'; fails. Instead import * as Rx from 'rxjs'; works
- `importD rxjs{,/Rx}`

(21) ⋆Dropping support for chaining operators (use pipe instead)
- `call (<rxjs{,/**/*}>?** \ <rxjs{,/**/*}/operators>)`

```
.{flatMap,map,mapTo,filter,take,takeUntil,takeWhile,delay
 ,last,null,do,catch,finally,switch,withLatestFrom,startWith
 ,timeout,defaultIfEmpty,scan,debounceTime,throttleTime
 ,share,mergeMap,retryWhen,timestamp,mergeMapTo
 ,distinctUntilChanged,switchMap,bufferToggle,concatMap
 ,windowWhen,concatAll,first,toArray,isEmpty,mergeAll
 ,groupBy,reduce,buffer}
```

- `import rxjs/add{,/**/*}`

(22) ★Operator renames: do -> tap, catch -> catchError, switch -> switchAll, finally -> finalize, throw -> throwError
- `read <rxjs{,/**/*}>?**.{do,catch,switch,finally}`
- `import rxjs/**/throw`
- `read <rxjs{,/**/*}>.Observable.throw`

(23) ★rxjs@6.0.0 is incompatible with the npm package symbol-observable
- `importD symbol-observable`

core-js 3.0.0

(1) update: asap (old stage 0 proposal) replaced by queueMicrotask (a part of HTML spec)
- `import core-js/**/*asap`
- `read {<core-js>,<global>}.asap`

(2) update: Update Observable (#257, #276, etc.)
- `call <core-js/**/*observable>`

(3) update: Update Array#flatten -> Array#flat and Array#flatMap
- `import core-js/**/flatten`
- `read {<core-js>?,<Array>?}**.flatten`

(4) update: Update String#matchAll (proposal-string-matchall#17, proposal-string-matchall#38, proposal-string-matchall#41, etc.) and move to the stage 3
- `call {<core-js>?,<String>?}**.matchAll`

(5) update: Update .name properties of String#{trimStart, trimEnd , trimLeft, trimRight}, move to the stage 3
- `import core-js/**/*{trim-left,trim-right}`

(6) remove obsolete: Error.isError (withdrawn)
- `import core-js/**/is-error`
- `read <Error>.isError`

(7) remove obsolete: System.global and global (replaced by globalThis)
- `read <System>.global`

(8) remove obsolete: Map#toJSON and Set#toJSON (rejected)
- `read {<Set>,<Map>}?**.toJSON`

(9) remove obsolete: RegExp.escape (rejected)
- `import core-js/**/escape`
- `read <RegExp>.escape`

(10) remove obsolete: Reflect.enumerate (removed from the spec)
- `read <Reflect>.enumerate`

(11) remove: Dict
- `call <Dict>`

(12) remove: Object.{classof, isObject, define, make}
- `import core-js/**/object/{classof,is-object,define,make}`
- `read <Object>.{classof,isObject,define,make}`

(13) remove: Function#part
- `import core-js/**/part`
- `read {<core-js>?,<Function>?}**.part`

(14) remove: String#{escapeHTML, unescapeHTML}
- `import core-js/**/*{escape-html,unescape-html}`
- `read <String>?**.{escapeHTML,unescapeHTML}`

(15) remove: delay
- `import core-js/**/delay`
- `readO <core-js/**/>.delay`
- `call {<core-js/**/>.delay,<core-js/**/delay>}`

(16) Package-related: Leave only one pair of bundles (global, with all polyfills) and move it to core-js-bundle package.
- `import core-js/{client,client/**/*}`

(17) Commonjs API/namespace: Move core-js/library to separate core-js-pure package.

- `import core-js/library/{es5,es6,es7}/*`

(18) Commonjs API/namespace: Because of removing all non-standard features, we no longer need core-js/shim entry point, replace it just with core-js.
- `import core-js/{shim,shim/**/*}`

(19) Commonjs API/namespace: Move all features from ES5, ES2015, ES2016, ES2017, ES2018 and ES2019 to one namespace for stable ES - it's available as core-js/es, all those features in modules folder has es. prefix.
- `import core-js/{es5,es6,es7,es5/**/*,es6/**/*,es7/**/*}`
- import core-js/modules/

```
{es6.array.copy-within,es6.array.every,es6.array.fill
,es6.array.filter,es6.array.find-index,es6.array.find
,es6.array.for-each,es6.array.from,es6.array.index-of
,es6.array.is-array,es6.array.iterator,es6.array.join
,es6.array.last-index-of,es6.array.map,es6.array.of
,es6.array.reduce-right,es6.array.reduce,es6.array.slice
,es6.array.some,es6.array.sort,es6.array.species,es6.date.now
,es6.date.to-iso-string,es6.date.to-json,es6.date.to-primitive
,es6.date.to-string,es6.function.bind,es6.function.has-instance
,es6.function.name,es6.map,es6.math.acosh,es6.math.asinh
,es6.math.atanh,es6.math.cbrt,es6.math.clz32,es6.math.cosh
,es6.math.expm1,es6.math.fround,es6.math.hypot,es6.math.imul
,es6.math.log1p,es6.math.log2,es6.math.log10,es6.math.sign
,es6.math.sinh,es6.math.tanh,es6.math.trunc,es6.number.constructor
,es6.number.epsilon,es6.number.is-finite,es6.number.is-integer
,es6.number.is-nan,es6.number.is-safe-integer
,es6.number.max-safe-integer,es6.number.min-safe-integer
,es6.number.parse-float,es6.number.parse-int
,es6.number.to-fixed,es6.number.to-precision,es6.object.assign
,es6.object.create,es6.object.define-properties
,es6.object.define-property,es6.object.freeze
,es6.object.get-own-property-descriptor
,es6.object.get-own-property-names,es6.object.get-prototype-of
,es6.object.is-extensible,es6.object.is-frozen,es6.object.is-sealed
,es6.object.is,es6.object.keys,es6.object.prevent-extensions
,es6.object.seal,es6.object.set-prototype-of,es6.object.to-string
,es6.parse-float,es6.parse-int,es6.promise,es6.reflect.apply
,es6.reflect.construct,es6.reflect.define-property
,es6.reflect.delete-property,es6.reflect.get-own-property-descriptor
,es6.reflect.get-prototype-of,es6.reflect.get,es6.reflect.has
,es6.reflect.is-extensible,es6.reflect.own-keys
,es6.reflect.prevent-extensions,es6.reflect.set-prototype-of
,es6.reflect.set,es6.regexp.constructor
,es6.regexp.exec,es6.regexp.flags,es6.regexp.to-string,es6.set
,es6.string.anchor,es6.string.big,es6.string.blink
,es6.string.bold,es6.string.code-point-at,es6.string.ends-with
,es6.string.fixed,es6.string.fontcolor,es6.string.fontsize
,es6.string.from-code-point,es6.string.includes
,es6.string.italics,es6.string.iterator,es6.string.link
,es6.string.raw,es6.string.repeat,es6.string.small
,es6.string.starts-with,es6.string.strike,es6.string.sub
,es6.string.sup,es6.string.trim,es6.symbol
,es6.typed.array-buffer,es6.typed.data-view
,es6.typed.float32-array,es6.typed.float64-array
,es6.typed.int8-array,es6.typed.int16-array
,es6.typed.int32-array,es6.typed.uint8-array
,es6.typed.uint8-clamped-array,es6.typed.uint16-array
,es6.typed.uint32-array,es6.weak-map,es6.weak-set
,es7.array.flat-map,es7.array.flatten,es7.array.includes
,es7.object.define-getter,es7.object.define-setter
```

```
            ,es7.object.entries,es7.object.get-own-property-descriptors
            ,es7.object.lookup-getter,es7.object.lookup-setter
            ,es7.object.values,es7.promise.finally,es7.string.pad-end
            ,es7.string.pad-start,es7.string.trim-left
            ,es7.string.trim-right,es7.symbol.async-iterator}
```

(20) Commonjs API/namespace: Change prefix for ES proposals from es7. to esnext., they no longer available in core-js/es7, use core-js/stage/* instead of that.
- `import core-js/modules/`
```
            {es7.map.from,es7.map.of,es7.math.clamp
            ,es7.math.deg-per-rad,es7.math.degrees
            ,es7.math.fscale,es7.math.iaddh,es7.math.imulh
            ,es7.math.isubh,es7.math.rad-per-deg,es7.math.radians
            ,es7.math.scale,es7.math.signbit,es7.math.umulh
            ,es7.observable,es7.promise.try
            ,es7.reflect.define-metadata,es7.reflect.delete-metadata
            ,es7.reflect.get-metadata,es7.reflect.get-metadata-keys
            ,es7.reflect.get-own-metadata
            ,es7.reflect.get-own-metadata-keys
            ,es7.reflect.has-metadata,es7.reflect.has-own-metadata
            ,es7.reflect.metadata,es7.set.of,es7.string.at
            ,es7.string.match-all,es7.symbol.observable
            ,es7.weak-map.from,es7.weak-map.of
            ,es7.weak-set.from,es7.weak-set.of}
```

(21) Commonjs API/namespace: Rename core-js(/library)/fn to core-js(-pure)/features for improve readability.
- `import core-js/{,library/}fn{,/**/*}`

(22) Commonjs API/namespace: Rename web.dom namespace to web.dom-collections.
- `import core-js/modules/web.dom.*`

(23) Commonjs API/namespace: Relax /modules/ directory by moving internal modules to /internals/ directory.
- `import core-js/modules/_*`

(24) Commonjs API/namespace: Remove deprecated array entry points: core-js(/library)/fn/array/{pop, push, reverse, shift, unshift}.
- `import core-js{/library/,/}fn/array/{pop,push,reverse,shift,unshift}`

(25) Commonjs API/namespace: core object no longer available in the global version, entry points which previously returned it now returns globalThis object. Also, don't set global core property.
- `read <global>.core`

yargs 14.0.0

(1) do not allow additional positionals in strict mode
- `call <yargs>**.strict [0, 0]`

node-fetch 2.0.0

(1) Major: require('node-fetch/lib/response') etc. is now unsupported; use require('node-fetch').Response or ES6 module imports
- `import node-fetch/lib/**/*`

(2) Major: response.text() no longer attempts to detect encoding, instead always opting for UTF-8 (per spec); use response.textConverted() for the v1 behavior
- `call <node-fetch>?**.text`

(3) Major: make response.json() throw error instead of returning an empty object on 204 no-content respose (per spec; reverts behavior changed in v1.6.2)
- `call <node-fetch>?**.json`

(4) Major: internal methods are no longer exposed
- `call <node-fetch>.{_clone,_decode,_convert}`

(5) Major: throw error when a GET or HEAD Request is constructed with a non-null body (per spec)
- `call <node-fetch> [2, 2]`

(6) Major: remove headers.getAll(); make get() return all headers delimited by commas (per spec)
- `call <node-fetch>?**.getAll`

(7) Enhance: make sure header names and values are valid in HTTP
- `call <node-fetch>?**.{set,append,has,delete} [1, 2] 0:string`

(8) Enhance: add response.arrayBuffer() (also applies to Requests)
- `read <node-fetch>?**.arrayBuffer`

(9) ⋆request.url can no longer be written in a property write
- `write <node-fetch>?**.url`

winston 3.0.0

(1) winston.Logger has been replaced with winston.createLogger.
- `read <winston>.Logger`

(2) winston.setLevels has been removed. Levels are frozen at the time of Logger creation.
- `call <winston>.setLevels`

(3) winston.transports.Memory was removed. Use any Node.js stream.Writeable with a large highWaterMark instance instead.
- `read <winston>.transports.Memory`

(4) When writing transports use winston-transport instead of winston.Transport.
- `read <winston>.Transport`

(5) In winston.transports.Console, output for all log levels are now sent to stdout by default. - stderrLevels option now defaults to []. - debugStdout option has been removed.
- `call <winston>.transports.Console [0, 2]`

(6) winston.Container instances no longer have default Console transports
- `call <winston>.Container [0, 0]`
- `call <winston>.Container [1, 1]`

(7) winston.Container.prototype.add no longer does crazy options parsing. Implementation inspired by segmentio/winston-logger
- `read <winston>.Container()?**.add [2, 2]`

(8) winston.Logger.log and level-specific methods (.info, .error, etc) no longer accepts a callback. The vast majority of use cases for this feature was folks awaiting all logging to complete, not just a single logging message. To accomplish this:
- `call <winston>?**.{info,warn,error} [2, 2] 1:function`
- `call <winston>?**.log [3, 3] 2:function`

(9) winston.Logger.add no longer accepts prototypes / classes. Pass an instance of our transport instead.
- `call <winston>?**.add [1, 2] 0:function`

(10) Logger.prototype.stream - options.transport is removed. Use the transport instance on the logger directly.
- `call <winston>?**.stream [1, 1]`

(11) Logger.prototype.query - options.transport is removed. Use the transport instance on the logger directly.
- `call <winston>?**.query [2, 2] 1:function`

(12) winston.exception has been removed. Use: const exception = winston.ExceptionHandler();
- `read <winston>.exception`

(13) winston.hash was removed.
- `read <winston>.hash`

(14) winston.common.pad was removed.
- `read <winston>.common.pad`

(15) winston.common.serialized was removed (use winston-compat).
- `read <winston>.common.serialize`

(16) winston.common.log was removed (use winston-compat).
- `read <winston>.common.log`

(17) Removed winston.transports.{File,Console,Http} formatting options
- `call <winston>.transports.{File,Console,Http} [1, 1]`

(18) Migrating filters and rewriters to formats in winston@3
- `read <winston>?**.{rewriters,filters}`

(19) ⋆Removed console property from transports
- `read <winston>?**.console`

(20) ⋆winston.config.colorize is removed
- `read <winston>.config.colorize`

redux 4.0.0

(1) Throw if getState, subscribe, or unsubscribe called while dispatching (including inside a reducer) (#1569 by @mjw56)
- `call <redux{/**/*,}>?**.{getState,subscribe}`

(2) Bundle cjs and es formats (#2358 by @TrySound) - (direct, private imports (import createStore from 'redux/lib/createStore') will no longer work)
- `import redux/**/*`

jsonwebtoken 8.0.0

(1) JOI validation library has been removed. The errors raised for input validation on sign function have changed, previously we raised ValidationError, now it is a plain Error. Error messages have also changed.
- `call <jsonwebtoken>.{sign,verify} [3, 4]`

(2) process.nextTick() execution before calling the verification callback has been removed. If you pass a callback function it will get called right away, if you expected the next tick behavior, you may want to call nextTick in your code.
- `call <jsonwebtoken>.{sign,verify} [4, 4]`

(3) maxAge verification option expects seconds or a string describing a time span (i.e: '2 days'). It accepted milliseconds in v7, which did not make sense since JWTs times are expressed in seconds.
- `call <jsonwebtoken>.verify [3, 4]`

(4) Unintentionally removed support for Streams on payload for signing.
- `call <jsonwebtoken>.sign [2, 4]`

mongoose 5.0.0

(1) BREAKING CHANGE: always use mongoose aggregation cursor when using .aggregate().cursor() #5941
- `write <mongoose>?**.cursor`

(2) BREAKING CHANGE: attach query middleware when compiling model #5939
- `call <mongoose>?**.{pre,post} [2, 2] 0:string 1:function`

(3) BREAKING CHANGE: remove passRawResult option for findOneAndUpdate, use rawResult #5869
- `write <mongoose>?**.passRawResult`

(4) BREAKING CHANGE: implicit async validators (based on number of function args) are removed, return a promise instead #5824
- `call <mongoose>?**.validate 0:function2`

(5) BREAKING CHANGE: mapReduce resolves to an object with 2 keys rather than 2 separate args #5816
- `call <mongoose>?**.mapReduce`

(6) BREAKING CHANGE: mongoose.connect() returns a promise, removed MongooseThenable #5796
- `callR <mongoose>.connect`
- `callR <mongoose>.createConnection [1, 3]`

(7) BREAKING CHANGE: query stream removed, use cursor() instead #5795
- `call <mongoose>?**.stream [0, 1]`

(8) BREAKING CHANGE: connection open() and openSet() removed, use openUri() instead #5795
- `call <mongoose>?**.{open,openSet}`

(9) BREAKING CHANGE: remove support for $pushAll, remove usePushEach option #5670
- `write <mongoose>?**.usePushEach`
- `write <mongoose>?**.$pushAll`

(10) BREAKING CHANGE: remove saveErrorIfNotFound, always error out if save() did not update a document #4973
- `write <mongoose>?**.saveErrorIfNotFound`
- `call <mongoose>?**.save`

(11) BREAKING CHANGE: don't execute getters in reverse order #4835
- `call <mongoose>?**.get().get [1, 1] 0:function`

(12) BREAKING CHANGE: toObject() and toJSON() option parameter merges with defaults rather than overwriting #4131
- `call <mongoose>?**.{toObject,toJson} [1, 1]`

(13) BREAKING CHANGE: deleteX() and remove() promise resolves to the write object result #4013
- `callR <mongoose>?**.{remove,deleteOne,deleteMany}`

(14) BREAKING CHANGE: aggregate() no longer accepts a spread #2716
- `call <mongoose>?**.aggregate [1,] 0:object`